http://www.knom.or.kr/ ISSN : 2586-0232

2020
통신망운용관리 학술대회 논문집
Proceeding of KNOM Conference 2020

NETWORK SDN BLOCKCHAIN

일시: 2020년 5월 15일 (금)
장소: On-line KNOM Conference Venue
주관: 한국통신학회, 통신망운용관리연구회 (KICS KNOM)
주최: 한국과학기술정보연구원 (KISTI)

한국통신학회 통신망운용관리연구회

초대의 말씀

한국통신학회 통신망 운용관리연구회(KNOM)는 **2020년 통신망 운용관리 학술대회 (KNOM Conference 2020)**를 통하여 통신망 운용관리 기술의 최신 연구 개발 현황을 국내 관련 분야 학자, 연구원, 네트워크 관리자, 및 실무 담당자들에게 소개하고, 활발한 토론을 할 수 있는 장을 마련하고자 튜토리얼 및 연구 논문을 모집합니다.

네트워크와 컴퓨팅 기술은 최근 급속하게 발전하고 변화하고 있습니다. 날로 고속화되는 무선기술과 차세대 인터넷에 대한 지대한 관심은 이러한 네트워크 기술 발전을 대변하고 있습니다. 또한, 클라우드 컴퓨팅, 모바일 컴퓨팅 등 향후 컴퓨터의 사용 개념을 혁신할 새로운 컴퓨팅 기술이 각광을 받고 있습니다. 또한, 영상 데이터의 급속한 확산은 유무선을 포함한 모든 네트워크에서 상상하기 어려운 새로운 데이터 폭주로 이어지고 있으며 언제 어디서나 동영상을 볼 수 있는 컴퓨팅 체계가 마련되고 있습니다. 빠른 통신기술의 발전과 보급은 무선 Data Explosion이라는 새로운 문제를 야기하고 있고 이를 해결하는 것이 통신 사업자의 가장 큰 이슈가 되어, 지난 10여 년간 통신망 운용관리 분야의 주요 연구주제였던 End-to-End 네트워크 관리는 유무선 통합 네트워크 환경에서 네트워크와 서버 및 단말을 포함해 관리해야 하는 현실적인 문제로 대두되었습니다. 또한, 클라우드 컴퓨팅의 보편화는 네트워크 구조와 트래픽의 흐름을 근본적으로 바꾸어가고 있으며, 네트워크와 서비스에 대한 보안침해도 급격히 증가하여 통신망 운용관리 분야의 연구와 개발 범위 또한 급속히 넓어지고 그 중요성이 더욱 강조되고 있습니다.

이러한 추세를 반영하여 **KNOM Conference 2020**에서는 통신망 전반에 대한 모델링, 설계, 서비스 제공, 운용 관리 및 보안 기술 분야의 최신 연구 개발 결과에 대한 유익한 정보제공과 토론의 장을 제공할 계획입니다. 부디 본 Conference가 운용관리 전반에 걸친 활발한 기술교류와 토론의 장이 될 수 있도록 각 분야의 전문가 여러분들의 적극적인 논문투고와 발표를 기대합니다.

또한, 본 학술대회에 통신망 운용관리와 관련된 연구소, 학계, 통신서비스 사업자, 통신기기 제조업체 등에서 많은 분이 참석하여 실제적인 기술을 습득하고 토론할 수 있는 좋은 기회가 될 수 있기를 바랍니다.

2020. 03. 02.

2020 통신망운용관리학술대회 운영위원장 석우진

한국통신학회 통신망운용관리연구회 위원장 주홍택

운영위원회

운영위원장	석우진(KISTI)	**홍보 및 출판**	주홍택(계명대)
학술	조부승(KISTI)	**등록 및 예산**	최미정(강원대)
자문	홍충선(경희대), 홍원기(POSTECH), 최덕재(전남대), 송왕철(제주대), 김영탁(영남대) 이영우(KT), 유재형(포항공대)		

프로그램

일시: 2020년 5월 15일 (금)
발표장소: KNOM 화상회의 시스템 (https://webinar.kafe.or.kr)

시간	주요행사	
09:10 - 09:30	**개 회 사**: KNOM2020 학술대회 위원장 석우진 센터장(KISTI) **축 사**: 통신망운영관리연구회(KNOM) 위원장 주홍택 교수(계명대) **Seminar Name**: KNOM2020-OPENNING	
	Track Ⅰ. 네트워크 관리 기술	**Track Ⅱ. 블록체인 및 보안 기술**
09:30 - 10:40	**TS1.** 5G (좌장: 조진용) **Seminar Name**: KNOM-TS01-5G	**TS2.** 블록체인 관리 Ⅰ (좌장: 문정훈) **Seminar Name**: KNOM-TS02-BLOCKCHAIN1
10:40 - 10:50	**Coffee Break**	
10:50 - 12:00	**TS3.** AI 기반 네트워크 관리 (좌장: 석우진) **Seminar Name**: KNOM-TS03-AI	**TS4.** 블록체인 관리 Ⅱ (좌장:장민석) **Seminar Name**: KNOM-TS04-BLOCKCHAIN2
12:00 - 13:30	**Lunch**	
13:30 - 14:40	**TS5.** SDN/NFV (좌장: 조진용) **Seminar Name**: KNOM-TS05-SDN	**TS6.** 블록체인 관리 Ⅲ (좌장: 손일권) **Seminar Name**: KNOM-TS06-BLOCKCHAIN3
14:40 - 14:50	**Coffee Break**	
14:50 - 16:00	**TS7.** 클라우드 및 가상 네트워크 관리 (좌장: 석우진) **Seminar Name**: KNOM-TS07-CLOUD	**TS8.** IoT/유무선 네트워크 관리 (좌장: 문정훈) **Seminar Name**: KNOM-TS08-IOT
16:00 - 16:10	**Coffee Break**	
16:10 - 17:20	**TS9.** 네트워크 응용 (좌장:장민석) **Seminar Name**: KNOM-TS09-APPLICATION	**TS10.** 네트워크 보안 (좌장: 손일권) **Seminar Name**: KNOM-TS10-SECURITY
17:20 - 17:50	**폐회식** **Seminar Name**: KNOM2020-CLOSING	
18:00 -	만찬	

논문 목차

Technical Session 1. 5G

Technical Session 2. 블록체인 관리 I

Technical Session 3. AI 기반 네트워크 관리

Technical Session 4. 블록체인 관리 II

Technical Session 5. SDN/NFV

Technical Session 6. 블록체인 관리 III

Technical Session 7. 클라우드 및 가상 네트워크 관리

Technical Session 8. IoT/유무선 네트워크 관리

Technical Session 9. 네트워크 응용

Technical Session 10. 네트워크 보안

백업 슬라이스 기반 5G 구조에 관한 연구

고한얼, 이재욱, 백상헌
고려대학교

heko@korea.ac.kr, iioiioiio123@korea.ac.kr, shpack@korea.ac.kr

The Study on Backup Slice-Based 5G Architecture

Haneul Ko, Jaewook Lee, and Sangheon Pack
Korea Univ.

요 약

5G 네트워크에서는 하나의 물리적인 네트워크를 여러 개의 논리적인 네트워크로 나누는 기술인 네트워크 슬라이싱 기술을 통해 다양한 서비스를 효과적으로 지원할 수 있다. 하지만, 네트워크 슬라이스에 예상치 못한 트래픽 부하가 급증하였을 때, 해당 슬라이스에 할당된 한정적인 자원이 급증한 부하를 효과적으로 처리하지 못할 가능성이 존재한다. 본 논문에서는 이러한 문제를 해결하기 위해 여러 서비스가 공유할 수 있는 백업 슬라이스 기반의 5G 네트워크 슬라이싱 구조를 제안한다. 백업 슬라이스 기반의 5G 네트워크 슬라이싱 구조의 성능 평가를 위해 평균 시스템 봉쇄확률과 시스템 이용률을 큐잉이론을 이용하여 도출하고, 기존 슬라이싱 구조와의 성능 비교를 통해 본 논문에서 제시한 구조의 우수성을 입증한다.

Ⅰ. 서 론

5G 네트워크에서 서비스 공급자는 네트워크 슬라이싱 기술을 통해 하나의 물리적 망을 가지고 특정 서비스에게 제공할 수 있는 다수의 네트워크 슬라이스를 구현한다. 즉, 각각의 네트워크 슬라이스는 가상 네트워크이며, 모든 슬라이스는 하나의 물리 네트워크에 매핑 된다. 이러한 기술을 통해 서비스 공급자는 다양한 서비스를 효과적으로 지원할 수 있다. 하지만, 네트워크 슬라이스에 트래픽 부하가 급증하였을 때, 해당 슬라이스에 할당된 자원들이 급증한 부하를 효과적으로 처리하지 못할 가능성이 존재한다. 본 논문에서는 이러한 문제를 해결하기 위해 여러 서비스가 공유할 수 있는 백업 슬라이스 기반의 5G 네트워크 슬라이싱 구조를 제안한다. 백업 슬라이스 기반의 5G 구조의 성능 평가를 위해 평균 시스템 봉쇄확률과 시스템 이용률을 큐잉이론을 통해 도출하고 기존 슬라이싱 구조와의 성능 비교를 통하여 본 논문에서 제시한 구조의 우수성을 입증한다.

Ⅱ. 백업 슬라이스 기반의 5G 구조

본 장에서는 네트워크 슬라이싱 서비스의 고가용성을 높이고자 백업 슬라이스 개념을 제안하고, 해당 백업 슬라이스 성능의 정성적 평가를 분석한다.

5G 이동통신 시스템에서는 네트워크 슬라이싱을 통해 다양한 서비스를 제공할 수 있을 것으로 기대된다. 하지만, 네트워크 슬라이싱 역시 실제 물리적 자원 위에 가상으로 자원을 할당하여 여러 서비스가 동시에 사용하는 것이기 때문에 자원 할당 및 관리 문제가 발생한다. 특히, 한 슬라이스에서 자원이 부족해서 추가 자원을 요청할 경우, 다른 슬라이스의 성능 저하 문제가 발생할 수 있다. 이는 네트워크 슬라이싱의 기본 모토인 슬라이스 간의 고립 (즉, isolation)을 위반하는 것이다. 이를 방지하고자 본 연구에서는 백업 슬라이스를 도입하여 위의 문제를 해결하고자 한다.

그림 1 은 백업 슬라이스가 포함된 네트워크 슬라이스를 기반으로 하는 5G 네트워크 구조를 나타낸다. 각 슬라이스는 eMBB, mMTC 등과 같은 서로 독립된 서비스를 제공한다. 이 때, 해당 서비스를 사용하는 사용자가 급증하는 경우, 해당 슬라이스가 급증하는 트래픽을 수용하지 못하는 문제가 발생할 수 있다. 이 경우에 모든 서비스를 수용할 수 있는 백업 슬라이스가 해당 트래픽을 수용하도록 한다.

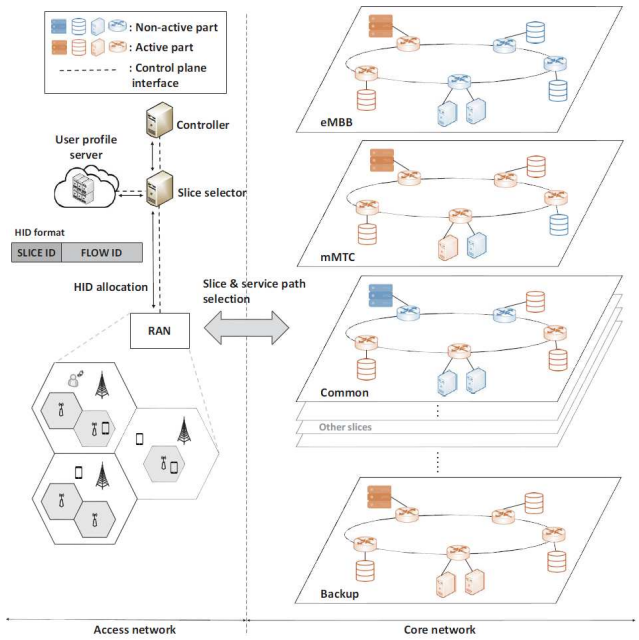


그림 1. 백업 슬라이스 기반 5G 구조

본 연구에서 제시한 백업 슬라이스의 성능 평가를 수학적 모델링을 통해 진행한다. 각 슬라이스를 M/M/1/K 큐라고 생각했을 경우, 시스템의 봉쇄 확률은 다음과 같이 계산된다.

우선, P_s 는 슬라이스 s 의 봉쇄 확률이다. 즉, 백업 슬라이스는 P_s 의 확률로 슬라이스 s 가 수용하지 못한 트래픽을 수용한다 (백업슬라이스는 $(1-P_s)$ 의 확률로 사용되지 않는다). 한편, $\mathbf{S_O}$ 와 $\mathbf{S_L}$ 이 과부화된 슬라이스와 과부화 되지 않은 슬라이스의 집합, 그리고 집합 $\mathbf{S_O}$ 에 대한 백업 슬라이스의 봉쇄 확률이 $P_b^{S_O}$ 이라고 할 때, 시스템의 평균 봉쇄확률은 다음과 같이 계산된다.

$$P_b = \sum_{\forall \mathbf{S_O}} \left[\prod_{s \in \mathbf{S}_L} (1 - P_s) \times \prod_{s \in \mathbf{S_O}} P_s \times P_b^{\mathbf{S_O}} \right]$$

또한, 시스템의 평균 봉쇄확률과 유사한 방법으로 시스템의 이용률을 계산하면 다음과 같다.

$$U_b = \sum_{\forall \mathbf{S_O}} \left[\prod_{s \in \mathbf{S}_L} (1 - P_s) \times \prod_{s \in \mathbf{S_O}} P_s \times (1 - P_b^{\mathbf{S_O}}) \times \rho_b^{\mathbf{S_O}} \right]$$

III. 성능 평가

백업 슬라이스 구조를 평가하기 위해, 기존의 5G 슬라이스 구조와 성능비교를 수행하였다. 3 개의 네트워크 슬라이스가 있는 경우를 가정하고 성능평가를 진행하였다. **오류! 참조 원본을 찾을 수 없습니다.**2 는 3 번째 슬라이스에서 처리해야하는 서비스 플로우가 들어오는 속도 (즉, λ_3)가 증가할 때, 시스템의 봉쇄확률 변화를 나타낸다.

오류! 참조 원본을 찾을 수 없습니다.2 에서 볼 수 있듯이 λ_3 가 증가할수록 기존 방법 및 본 연구에서 제안한 방법의 봉쇄확률이 증가하는 것을 확인할 수 있다. 하지만, λ_3 가 클 경우 본 연구에서 제안한 방법의 봉쇄확률이 더 낮은 것을 확인할 수 있다. 이는 다른 백업 슬라이스에서 3 번째 슬라이스에서 처리하지 못하는 서비스들을 수용하기 때문이다.

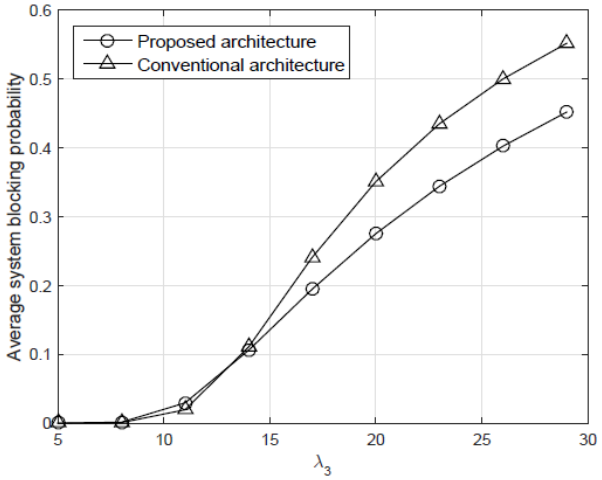


그림 2. 플로우 속도 변화에 따른 시스템 봉쇄확률

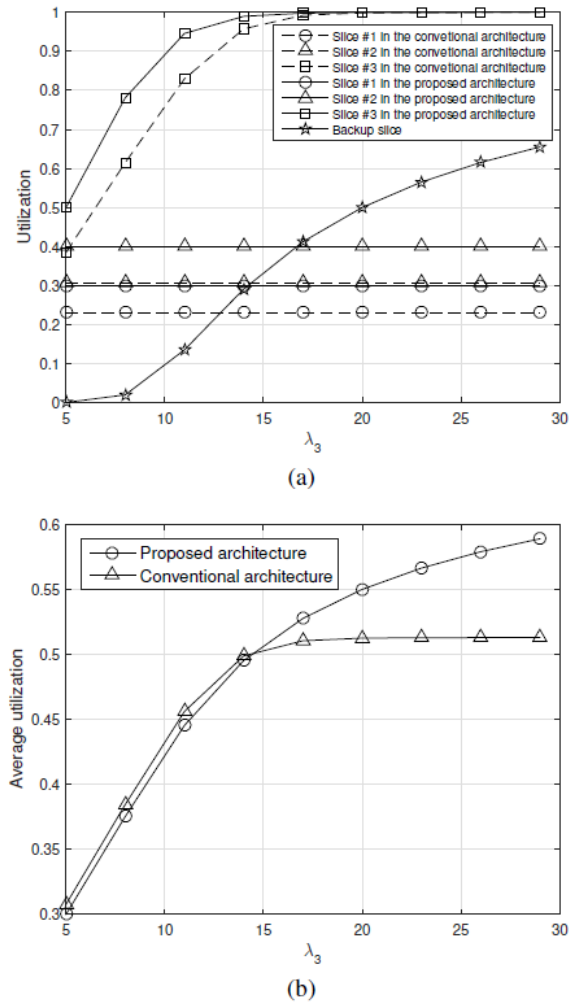


그림 3. 플로우 속도 변화에 따른 시스템 이용률

그림 3 은 플로우 속도 변화에 따른 시스템 이용률을 보여준다. 플로우 속도가 낮을 경우에는 본 논문에서 제시한 방법의 이용률이 낮게 나타나지만 속도가 일정 수준 이상이 되었을 경우에는 이용률이 높아지는 것을 볼 수 있다. 이는 플로우 속도가 낮을 때는 백업 슬라이스에 할당되어 있는 자원이 단순히 낭비될 수 있지만 플로우 속도가 높아짐에 따라 백업 슬라이스에 할당된 자원이 적절하게 사용되어 전체 시스템 이용률이 높아지기 때문이다.

IV. 결론

본 논문에서는 급격한 트래픽 부하 변화에도 유연하게 대응할 수 있는 백업 슬라이스 기반의 5G 구조를 제안하였다. 백업 슬라이스 기반의 5G 구조의 성능 평가를 위해 평균 시스템 봉쇄확률과 시스템 이용률을 큐잉이론을 이용하여 도출하고 기존 슬라이싱 구조와의 성능 비교를 통하여 본 논문에서 제시한 구조의 우수성을 입증하였다.

ACKNOWLEDGMENT

본 연구는 2019 년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(과제번호: 2019R1C1C1004352)

Ensuring Quality-of-Service (QoS) in eMBB-URLLC Co-existence Scenario

Yan Kyaw Tun and Choong Seon Hong
Department of Computer Science and Engineering, Kyung Hee University, Yongin-si, Gyeonggi-do 17104, Korea
email:{ykyawtun7, cshong}@khu.ac.kr

Abstract

In this work, we propose a resource scheduling optimization problem to ensure quality-of-service (QoS) in a dynamic multiplexing scenario of URLLC (Ultra-reliable low latency communication) and eMBB (Enhanced mobile broadband) services. The services requiring high bandwidth such as augmented reality and video streaming are classified as eMBB traffic, whereas services like remote surgery and autonomous driving that demand sub-millisecond latency with minimum error rates are classified as URLLC traffic in 5G New Radio (NR). We characterized these services for a dynamic multiplexing scenario, where we target to ensure the stringent requirements of URLLC traffics while guaranteeing the data rate of eMBB users. To that end, we formulate a resource scheduling optimization problem with multiplexed eMBB-URLLC services. Here, we consider a puncturing technique that allows URLLC traffic to schedule over the ongoing eMBB transmissions. Next, we investigate the problem of Resource Blocks (RBs) allocation to eMBB users with a 2-Dimensions Hopfield Neural Networks (2D-HNN) formulation. Simulation results show the efficacy of our proposed method, in terms of fairness and achievable data rate.

Keywords – *5G New Radio (NR), eMBB, URLLC, resource scheduling, Neural Networks.*

I. Introduction

The upcoming 5G networks expects unprecedented surge of heterogeneous cellular services. Specifically, the services requiring stringent latency requirements, and those demanding high data rate will dominate the upcoming mobile networks. However, in the existing network, it is almost impossible to ensure both of these requirements at the same time [1]. This is due to the fact that the current network architecture primarily focuses on maximizing the overall network throughput, using long packets. In doing so, to ensure ultra-reliability, a way would be to use short packets [2]; however, it dramatically reduces the data rate. Thus, the challenge to efficiently manage radio resources for fulfilling QoS requirements of the these diverse upcoming services remains.

In 5G New Radio (NR), new features classifying these peculiar services are defined based on their requirements [3]. The services are characterized as: (i) enhanced Mobile Broad Band (eMBB), (ii) massive Machine Type Communications (mMTC), and (iii) Ultra Reliable Low Latency Communications (URLLC). The services requiring high bandwidth such as augmented reality and video streaming are classified as eMBB traffic. Basically, eMBB is characterize like internet access service, similar to an extension to Long Term Evolution-Advanced (LTE-A) [4]. The services characterizing sporadic nature of traffic, such as with Internet-of-Things (IoT), in particular, like sensing and monitoring services, is defined as mMTC. It can be considered as narrow-band internet access where the nodes are active for a short time interval. Similarly, services like remote surgery and autonomous driving that demand sub-millisecond latency with minimum error rates are classified as URLLC traffic. For an example, reliability is defined as $(1-10^{-5})$ success probability while a user transmits a Protocol Data Unit (PDU) of 32 bytes within 1ms [5]. This is defined as QoS requirements of URLLC by the current 3GPP standards.

In this work, we employ puncturing techniques, which is one of the approach to deal with the stringent requirements (i.e., in terms of latency and reliability) for spectrum management in 5G NR to satisfy URLLC [6]. This method is concurrent to the standards set by 3GPP report, where URLLC traffic needs to be immediately transmitted for meeting their QoS requirements [7]. We consider the coexistence of eMBB-URLLC traffic, where under the puncturing mechanism, the 5G NodeB (gNB) will drop ongoing eMBB transmissions to satisfy QoS requirements of URLLC traffic in the upcoming time slot. The dropped eMBB users because of puncturing will be rescheduled [8], [9], [10]. To that end, we formulated a resource scheduling optimization problem with multiplexed eMBB-URLLC services. We adopted our proposed 2-Dimensional Hopfield Neural Network (2D-HNN) solution [11] to solve the formulated problem. In doing so, we use Cumulative Distribution Function (CDF) of the random URLLC traffic to relax the chance constraint to a deterministic linear constraint in the optimization problem.

II. Problem Formulation

We sequentially tackle the problem of resource scheduling and allocation in this section. In doing so, we first define the RB allocation formulation to derive an eMBB scheduler. Next, we consider the multiplexing scenario, where we consider the QoS constraint and latency requirements of both eMBB-URLLC services. Here, a neural network based eMBB users scheduling strategy is adopted, following our earlier work [11]. Furthermore, also relax the chance constraint to derive solutions for the scheduling problem.

We consider $K \in \mathcal{K}$ eMBB users available within a time-slot T and a set of URLLC nodes $\mathcal{U} \subseteq \mathcal{K}$ requesting services in frequency-time slot. B is the effective bandwidth in each slot, which is further divided into mini slots (t_m) such that resource $f_\eta(t_m) = B/\eta$, for $\eta > 0$. Furthermore, $f_\eta(t_m)$ is quantize into N levels. Then, $x_{k(t_m),N} \in \{0,1\}$ denotes the RB association variable such that $f_\eta(t_m) = x_k^N(t_m) \cdot N$, for each associated user k.

A. Resource Blocks Allocation to eMBB Traffics

The instantaneous rate for an eMBB user k at time slot T is given as

$$R_k^e = \sum\nolimits_{N\in\mathcal{N}} f_\eta(t_m) x_k^N(t_m) x_{k(t_m),N} \log_2\left(1 + \frac{\rho_k |G_k|^2}{N_o}\right), \quad \forall N \in \mathcal{N}, \tag{1}$$

where $f_\eta(t_m)x_k^N(t_m)x_{k(t_m),N}$ is the total spectrum allocated, ρ_k is the transmission power, $|G_k|^2$ is the channel gain between user k and the base station, and N_o is the noise power. Then, following (1), the rate of all eMBB users in time slot T is

$$R_k^e(T) = \sum\nolimits_{k\in\mathcal{K}} \sum\nolimits_{N\in\mathcal{N}} f_\eta(t_m) x_k^N(t_m) x_{k(t_m),N} \cdot \log_2\left(1 + \frac{\rho_k |G_k|^2}{N_o}\right), \ \forall k \in \mathcal{K}, N \in \mathcal{N} \tag{2}$$

such that $\sum_{k\in\mathcal{K}} f_\eta(t_m) x_k^N(t_m) x_{k(t_m),N} \leq B$.

In this regards, we can define the average data rate for user k up to time t as

$$\tilde{R}_k^e(t) = \zeta \tilde{R}_k^e(t-1) + (1-\zeta) R_k^e(t) \tag{3}$$

where $\zeta \in [0,1]$. Therefore, the optimization problem for eMBB scheduler can be formulated as

$$\begin{aligned} \underset{\mathrm{x}}{\text{Max}} \quad & \sum\nolimits_{k\in\mathcal{K}} \frac{R_k^e(T)}{[\tilde{R}_k^e(T)]^\alpha} \\ \text{s.t} \quad & C_1 : \sum\nolimits_{N\in\mathcal{N}} x_{k(t_m),N} \leq 1, \ \forall k \in \mathcal{K} \\ & C_2 : x_{k(t_m),N} \in \{0,1\}, \ \forall k \in \mathcal{K}, N \in \mathcal{N}. \end{aligned} \tag{4}$$

where the constraint (4)C_1 ensures that each RB is allocated to only one user at a time. The solution of (4) is the allocation matrix x with each element defined as

$$x_{k(t_m),N} = \begin{cases} 1, & \text{if } N \in \mathcal{N}_k; \\ 0, & \text{Otherwise}, \end{cases} \tag{5}$$

where $\mathcal{N}_k$ is the set of all RBs allocated to the eMBB user k. For shorthand representation, we will use $x_{k,N}$ for $x_{k(t_m),N}$ hereafter.

B. Resource Scheduling Mechanism for eMBB-URLLC Traffics

In a scenario of dynamic multiplexing between eMBB-URLLC traffics, we assume that the impact of eMBB traffic to punctured resources by URLLC traffic is proportional [12]. Consider η_u^k is the level of punctured RBs within a time slot T of eMBB user k i.e., the impact on the data rate of eMBB users.

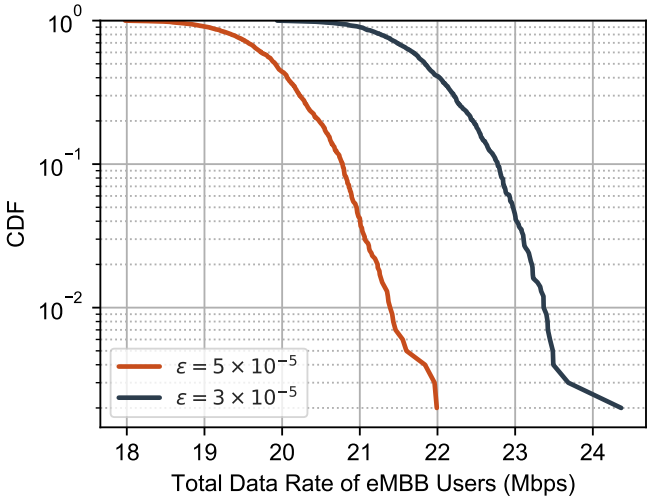


Fig. 1. CDF of the total data rate at $\alpha = 1$.

Let the arriving URLLC traffic load during time t is characterized by the random variable $X(t)$. Then, the corresponding outage probability of URLLC traffic is defined as

$$P(O) = P(R_T^u < X(t)) \tag{6}$$

where R_T^u is the instantaneous rate of URLLC traffic i.e.,

$$P(O) = P\left[\sum\nolimits_{k\in\mathcal{K}} \eta_u^k \log_2\left(1 + \frac{\rho_u |G_u|^2}{N_o}\right) < X(t)\right]. \tag{7}$$

Therefore, the proposed scheduler aims at maximizing the total data rate of eMBB users while satisfying the latency stringent constraint of URLLC traffic as follows:

$$\begin{aligned} \underset{\eta_u}{\text{Max}} \quad & \sum\nolimits_{k\in\mathcal{K}} \sum\nolimits_{N\in\mathcal{N}} \left(f_\eta(t_m) x_k^N(t_m) x_{k,N} - \eta_u^k\right) \\ & \cdot \log_2\left(1 + \frac{\rho_k |G_k|^2}{N_o}\right) \\ \text{s.t} \quad & C_1 : P(R_T^u < X) \leq \epsilon \\ & C_2 : \sum\nolimits_{k\in\mathcal{K}} \eta_u^k \leq f_\eta(t_m) x_k^N(t_m) x_{k,N}, \ \forall k \in \mathcal{K} \end{aligned} \tag{8}$$

where (8)C_1 characterizes the maximum outage probability of the URLLC traffic, namely the *reliability level* with the probability value ϵ. Constraint (8)C_2 ensures that the proportion of resources to the URLLC load is no more than the allocated resources for eMBB users. In order to obtain a close form solution for the optimization problem, we first need to relax the chance constraint (8)C_1. Then, we solve the problem by using 2-Dimensions Hopefield Neural Networks (2D-HNN).

III. Simulation Results

We evaluate the performance of our proposed solution approach in terms of achieved data rate and fairness. For this, we consider 10 eMBB users with different channel states. In each time slot, we consider that 100 RBs are available, and each RB is 180 kHz. Note that 5G NR permits a large number configuration varying from 15kHz to 480kHz. We will present them accordingly in this section.

In Fig. 1, for the different values of reliability metrics ϵ at $\alpha = 1$, we evaluate the CDF of the total data rate of all eMBB

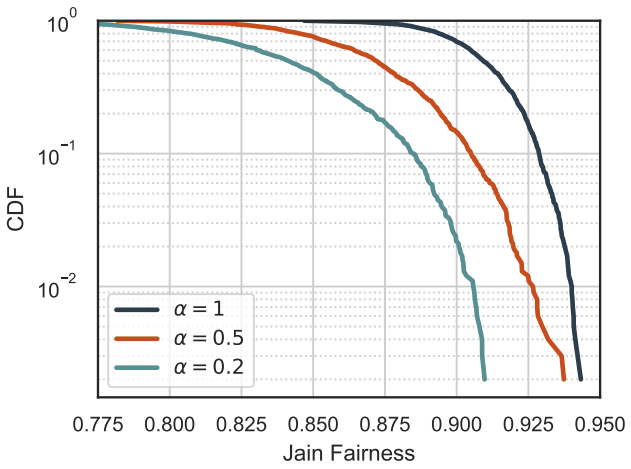


Fig. 2. Comparison of Fairness.

users. It is observed that a higher value of ϵ will limit the puncturing to satisfy URLLC reliability requirements. Thus, we see the increase in the overall sum rate for eMBB users. On the contrary, smaller values of ϵ means that we provide higher priority to the URLLC reliability constraint. Consequently, we observe the drop in the data rate of eMBB users as more resources are punctured.

Next, we use different values α and the reliability level of URLLC load ϵ, and evaluate the performance of the proposed mechanism following hopfield neural network solutions, as in [11]. We calculate the long-term data rate of all eMBB users and fairness among them for different parametric values α and ϵ. We use Jain's fairness index [13] to evaluate the fairness amongst the users which is given as

$$f(T_1, ..., T_z) = \frac{(\sum_{i=1}^{z} T_i)^2}{z \sum_{i=1}^{z} T_i^2}. \quad (9)$$

The value of Jain's fairness index lies within the interval [0, 1], i.e., around 0.95 for $\alpha = 1$, i.e, the system's fairness index is 0.95 which means that it is 95% fair).

IV. Conclusion

In this paper, we have studied a resource scheduling mechanism to ensure quality-of-service (QoS) in a dynamic multiplexing scenario of URLLC (Ultra-reliable low latency communication) and eMBB (Enhanced mobile broadband) services. Here, the scheduled eMBB traffic is punctured by URLLC traffic to fulfill its stringent latency requirements. In doing so, we have defined the RB allocation formulation to derive an eMBB scheduler that maximizes the overall sum-rate of eMBB users. Next, we have considered the multiplexing scenario, where we have the QoS constraint and latency requirements of both eMBB-URLLC services. We have solved the resource allocation problem with a modified 2D-HNN. In doing so, we have relaxed the chance constraint problem to a deterministic constraint using the CDF of the arrival URLLC traffic. Using the comparison results, we have demonstrated the efficacy of our proposed mechanism, where we have highlighted the impact of QoS constraints upon overall sum-rate of the network.

ACKNOWLEDGEMENT

This work was supported by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing) *Dr. CS Hong is the corresponding author.

References

[1] B. Soret, P. Mogensen, K. I. Pedersen, and M. C. Aguayo-Torres, "Fundamental tradeoffs among reliability, latency and throughput in cellular networks," in *Globecom Workshops (GC Wkshps), 2014*. IEEE, 2014, pp. 1391–1396.

[2] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

[3] "3gpp tsg ran wg1 88, tech. rep., february 2017," Tech. Rep.

[4] C. Hoymann, D. Astely, M. Stattin, G. Wikstrom, J.-F. Cheng, A. Hoglund, M. Frenne, R. Blasco, J. Huschke, and F. Gunnarsson, "Lte release 14 outlook," *IEEE Communications Magazine*, vol. 54, no. 6, pp. 44–49, 2016.

[5] P. Popovski, J. J. Nielsen, C. Stefanovic, E. De Carvalho, E. Strom, K. F. Trillingsgaard, A.-S. Bana, D. M. Kim, R. Kotaba, J. Park *et al.*, "Wireless access for ultra-reliable low-latency communication: Principles and building blocks," *Ieee Network*, vol. 32, no. 2, pp. 16–23, 2018.

[6] K. I. Pedersen, G. Pocovi, J. Steiner, and S. R. Khosravirad, "Punctured scheduling for critical low latency data on a shared channel with mobile broadband," in *Vehicular Technology Conference (VTC-Fall), 2017 IEEE 86th*. IEEE, 2017, pp. 1–6.

[7] Tech. Rep., study on New Radio Access Technology Physical Layer Aspects (Release 14).

[8] M. Alsenwi, N. H. Tran, M. Bennis, A. K. Bairagi, and C. S. Hong, "embb-urllc resource slicing: A risk-sensitive approach," *IEEE Communications Letters*, vol. 23, no. 4, pp. 740–743, 2019.

[9] S. R. Pandey, M. Alsenwi, Y. K. Tun, and C. S. Hong, "A downlink resource scheduling strategy for urllc traffic," in *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2019, pp. 1–6.

[10] M. Alsenwi, S. R. Pandey, Y. K. Tun, K. T. Kim, and C. S. Hong, "A chance constrained based formulation for dynamic multiplexing of embb-urllc traffics in 5g new radio," in *The International Conference on Information Networking (ICOIN 2019)*. IEEE, 2019, Kuala Lumpur, Malaysia.

[11] M. Alsenwi, I. Yaqoob, S. R. Pandey, Y. K. Tun, A. K. Bairagi, L.-w. Kim, and C. S. Hong, "Towards coexistence of cellular and wifi networks in unlicensed spectrum: A neural networks based approach," *IEEE Access*, vol. 7, pp. 110 023–110 034, 2019.

[12] A. Anand, G. de Veciana, and S. Shakkottai, "Joint scheduling of urllc and embb traffic in 5g wireless networks," *arXiv preprint arXiv:1712.05344*, 2017.

[13] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, 1984.

LTE-LAA 와 Wi-Fi 의 공존 시스템의 성능 개선에 관한 연구

이민정, 이재욱, 이호찬, 김태윤, 백상헌
고려대학교

{lmj1300, iioiioiio123, ghcks1000, kimtyoun123, shpack} @korea.ac.kr

A Study on the Performance Improvement for the Coexistence of LTE-LAA and Wi-Fi systems

Minjeong Lee, Jaewook Lee, Hochan Lee, Taeyun Kim, Sangheon Pack
Korea Univ.

요 약

최근 이동통신 기술의 발전과 함께 네트워크에 연결되는 새로운 단말의 수와 관련 서비스가 증가함에 따라 무선 트래픽을 감당하기 위한 수요가 증가하고 있다. 이러한 요구사항에 대응하기 위해 5GHz 비면허 대역에서 이동통신 기술을 지원하는 LTE-License Assisted Access (LAA) 가 3GPP 표준으로 제안되었다. 이에 따라 대표적인 비면허 대역 통신 기술인 LTE-LAA 와 Wi-Fi 기술의 효율적인 공존을 위해 Listen Before Talk (LBT) 기술에 대해 많은 연구가 이루어 지고 있다. 본 논문에서는 LTE-LAA 와 Wi-Fi 간 효율적인 공존을 위한 LBT 알고리즘 기법에 대해 살펴본다.

Ⅰ. 서 론

최근 이동통신 기술의 발전에 따라 IoT 장비의 확산과 같이 새로운 단말장비의 수가 증가하고 있다. 또한 모바일 미디어 소비증가와 같은 가입자의 이용형태의 변화로 가입자당 월간 트래픽이 매년 빠른 속도로 증가하고 있다[1].

이처럼 증가하는 트래픽과 향상된 Quality of Service (QoS) 요구 수준에 대응하기 위해 이동통신 사업자의 주파수 확보의 필요성이 대두되었다. 한정된 무선 주파수 자원의 특성상 사용가능한 주파수 대역을 넓히기 위해 5GHz 비면허 주파수 대역을 사용하는 LTE-Licensed-Assisted Access (LAA) 기술이 3GPP 표준으로 정의되었다[2]. LTE-LAA 는 5GHz 비면허 주파수 대역에서 사용되는 대표적인 통신 기술인 Wi-Fi 와 주파수 자원을 공유하여 사용할 때 공정성을 유지하기 위해 Listen Before Talk (LBT) 메커니즘을 사용한다.

LBT 메커니즘은 무선 채널에 신호를 전송하기 전에 채널이 유휴한지 확인하는 Clear Channel Assessment (CCA) 동작을 거친다. 이때, CCA 기간은 LTE-LAA 와 Wi-Fi 시스템들의 공존성에 큰 영향을 준다. CCA 기간이 긴 경우 LTE-LAA 시스템의 채널 사용률은 낮아지고, Wi-Fi 시스템의 채널 사용률은 높아진다. 이와 반대로, CCA 기간이 짧을수록 LTE-LAA 시스템의 채널 사용률이 높아지고, Wi-Fi 채널 사용률이 낮아진다.

따라서, 본 논문에서는 LTE-LAA 와 Wi-Fi 가 공존하는 네트워크 환경에서 LBT 알고리즘을 개선하여 LTE-LAA, Wi-Fi 의 전체 성능과 공정성을 높이는 기법에 대해 알아본다. Ⅱ장에서는 LBT 메커니즘에 대해 알아보고, Ⅲ장에서 LAA 와 Wi-Fi 의 공정성을 높이기 위한 기법인 Dual Threshold LBT[3]과 ReLBT[4]를 분석하며, Ⅳ장에서는 결론과 추후 연구방향에 대해 기술한다.

Ⅱ. LTE-LAA 의 LBT 메커니즘

비면허 주파수 대역에는 Wi-Fi 를 포함한 다양한 통신규격들이 존재하며 무선채널에 접속할 때 Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) 메커니즘을 사용한다. 3GPP Rel.13 LTE-LAA 표준에는 이에 대응하여 그림 1 과 같은 LBT 메커니즘을 정의하였다.

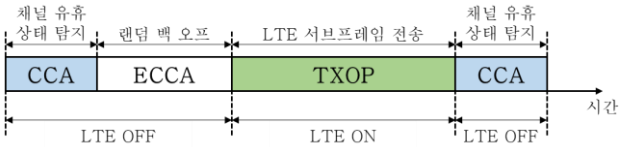


그림 1 LTE-LAA 의 LBT 메커니즘

LBT 동작은 무선구간의 충돌을 회피하기 위해 채널이 유휴한지 일정기간 확인한 후 전송하는 방식으로 동작한다. 3GPP 표준에서 정의한 LBT 메커니즘 중 LTE-LAA 와 Wi-Fi 간 공정성을 확보하고 전체 성능을 높일 수 있는 방법으로 Category 4 LBT (Cat 4 LBT)가 제안된다. Cat 4 LBT 동작은 크게 세 구간으로 구분되는데, 먼저 고정된 시간만큼 채널이 유휴상태인지 확인하는 CCA 동작을 거친다. CCA 동작 동안 채널이 혼잡할 경우 가변 기간동안 대기하는 Extended CCA (ECCA) 동작을 수행하고, 채널이 유휴 한 것으로 확인되면 LTE 다운링크 통신을 수행하는 Transmission Opportunity (TXOP) 동작을 수행한다.

ECCA 기간은 Contention Window (CW) 크기에서 랜덤 백 오프 값을 선택해 ECCA 기간이 결정된다. CW 의 크기는 TXOP 동안 Hybrid Automatic Repeat Request (HARQ)에서 Negative Acknowledgement (NACK)의 발생정도를 기반으로 증가하거나 감소하게 된다[2]. 하지만, 표준에서는 CW 크기를 결정하는 알고리즘에 대해 명시되지 않기 때문에, LTE-LAA 와 Wi-Fi 간 성능의 공정성을 보장하면서, 전체 성능을 최대화하는 알고리즘이 필요하다. 따라서, 다음 장에서는 최적의 CW 크기를 선택하는 기법들에 대해 분석한다.

Ⅲ. LBT 개선 기법

1) Dual Threshold LBT

Dual Threshold LBT 는 CCA 동안 채널이 혼잡한 횟수를 기반으로 2 가지 Threshold 를 사용하여 CW 크기를 조절한다.

먼저, 이전 CCA 구간 동안 Increase Threshold (iTH)에 도달할 경우 CW 크기를 2 배씩 증가시키고, Decrease Threshold (dTH)에 도달할 경우 CW 크기를 최소로 줄인다. iTH 는 LTE-LAA 가 무선채널을 사용하는 빈도를 줄이는 역할을 하고, dTH 는 CW 크기가 커진 상황에서도 최소한의 통신기회를 보장해 주는 역할을 한다. 최적의 Threshold 기준은 LTE-LAA 와 Wi-Fi 성능의 합이 크고, 성능의 차이가 적은 두가지 조건을 최대한 만족하는 값을 경사 하강법을 이용하여 계산한다.

본 기법은 User Equipment (UE)의 이동이나 출력 파워 등의 시스템 모델의 파라미터 변경이 있을 경우 최적의 Threshold 를 계산하기 위한 오프라인 연산이 요구된다.

2) ReLBT

ReLBT 는 표준에 정의된 HARQ 기반 충돌탐지 메커니즘 대신 충돌이 발생할 확률(P_{obs})을 계산하고, Q-러닝을 적용하여 CW 크기를 주어진 상황에 적합하게 반영하여 LTE-LAA 와 Wi-Fi 간 성능의 차이를 줄이는 LBT 메커니즘을 제안한다.

P_{obs}는 LBT 메커니즘의 CCA, ECCA, TXOP 기간 동안 관찰된 채널의 유휴상태, 혼잡상태와 NACK 를 통해 확인할 수 있는 전송실패 구간을 기반으로 계산한다.

$$CW_{cur} = \begin{cases} \min\left[2 \times CW_{pre} \times \omega^{p_{obs}},\ CW_{max}\right], \forall p_{obs} > 0 \\ \max\left[\dfrac{CW_{pre} \times \omega^{p_{obs}}}{2},\ CW_{min}\right], \forall p_{obs} = 0 \end{cases}$$

이렇게 계산된 P_{obs}를 기반으로 CW 크기를 변경하는 수식은 위와 같다. 가중치(ω)를 사용하여 ECCA 기간을 결정하는 과거 CW 크기(CW_{pre})를 다음에 사용할 CW 크기(CW_{cur})로 업데이트 한다. 늘어난 ECCA 기간만큼 LTE-LAA 가 패킷을 전송할 기회가 줄어드는데, ReLBT 는 Q-러닝을 적용하여 CW 크기를 보정하는 작업을 수행한다.

그림 2 의 ReLBT 의 Q-러닝 모델은 과거에 충돌가능성이 얼마나 있었는지를 상태로 사용하여 CW 크기를 더 늘릴지 줄일지를 결정하는 행동을 선택한다. 최종적으로 선택한 행동에 따라 업데이트 된 충돌이 발생하지 않을 확률을 보상으로 받는다.

ReLBT 는 두가지 CW 크기 조절 알고리즘을 사용하여 채널의 상태에 적합한 CW 크기를 선택하고 이를 통해 Wi-Fi 와 LTE-LAA 간 전송속도와 지연시간의 차이를 줄여 공정성을 개선하였다.

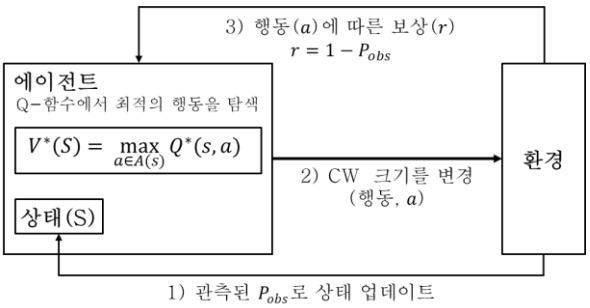


그림 2 ReLBT 의 Q-러닝의 동작 구조

Ⅳ. 결론

본 논문에서는 LTE-LAA 와 Wi-Fi 의 공존 시스템에서 성능 개선을 위한 기법 동향에 대하여 분석하였다. Dual Threshold LBT 기법은 무선환경의 변화가 있을 때 마다 다시 오프라인 연산을 진행해야 한다. ReLBT 의 경우 LTE-LAA 시스템이 Wi-Fi access Point(AP)와 정보를 교환하는 것이 불가능하므로, AP 의 급격한 동작 변화에 적합한 행동을 하지 못하는 경우가 발생할 수 있다. 향후에는 LTE-LAA 와 Wi-Fi 공존 시스템 환경에서 통신 사업자가 운영하는 각각의 Evolved Node B (eNB)와 AP 가 에이전트로 동작하는 멀티 에이전트 강화학습 모델을 사용하여 LBT 메커니즘의 효율성을 높이는 연구를 진행할 예정이다.

ACKNOWLEDGMENT

본 연구는 2020 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단 (중견과제)의 지원을 받아 수행된 연구이고(No. 2020R1A2C3006786), 또한 2017 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단 (전략과제)의 지원을 받아 수행된 연구임(No. 2017R1E1A1A01073742).

참 고 문 헌

[1] 장재혁, 박승근. “모바일 트래픽 동향,” ETRI 전자통신동향분석, 2019

[2] 3GPP TR 36.889 v13.0.0, “Feasibility Study on Licensed-Assisted Access to Unlicensed Spectrum,” 2015.07

[3] Jae-Hoon Kim and Sungchul Choi. “A Dual Threshold Listen-before-Talk for Unlicensed LTE Systems,” Eurasip Journal on Wireless Communications and Networking, vol. 2019, no. 1, 2019, pp. 1-14.

[4] Ali, Rashid, et al. “(ReLBT): A Reinforcement Learning-Enabled Listen before Talk Mechanism for LTE-LAA and Wi-Fi Coexistence in IoT,” Computer Communications, vol. 150, 2020, pp. 498-505.

UAV-HST 환경에서 강화학습 기반의 UAV 경로 최적화 연구

박유민, 홍충선*
경희대학교 컴퓨터공학과

yumin0906@khu.ac.kr, *cshong@khu.ac.kr

A Study on the Optimization UAV Trajectory based on Reinforcement Learning in Unmanned Aerial Vehicle-High Speed Train Environments

Yu Min Park, Choong Seon Hong*
Kyung Hee University

요 약

고속 열차에서 5G 기반의 VR/AR 의 컨텐츠를 사용자들이 경험하기 위해서는 새로운 통신 인프라가 필요하다. 따라서 일반적으로 열차의 통신을 지원하는 Rail-side Units(RSUs)를 대신하여 Unmanned Aerial Vehicle(UAV)를 통신 장비로써 활용하는 것을 제안한다. 하지만 UAV 보다 빠른 고속 열차와 통신을 안정적으로 유지하기 위해서는 고도, 속력, 방향 들을 최적으로 결정하여 경로를 이동해야 한다. 이에 본 논문은 고속 열차와 UAV 의 안정적인 통신을 위해 UAV 의 경로 최적화 기법을 제안한다. The Air-To-Ground Path Loss Model 을 토대로 가상 통신 환경을 조성하고 Unity Ml-agents 라이브러리를 통해 학습 환경을 구축한다. 본 논문은 UAV 경로 최적화를 위한 방법으로 강화학습 중 Soft-Actor-Critic(SAC)을 사용한다. 결과적으로 본 연구를 통해 고속 열차의 속도에 따른 UAV 의 최적 경로를 빠르고 정확하게 도출하여 통신을 안정적으로 유지할 수 있었다.

1. 서론

차세대 이동 통신인 5G 는 밀리미터파를 이용한 무선 네트워크 기술이다. 5G 는 대용량의 데이터를 초고속으로 무선 통신할 수 있게 하여 VR/AR 등 고사양의 컨텐츠를 사용자들이 무선으로 경험할 수 있게 하였다. 하지만 5G 는 사용하는 밀리미터 파의 직진성으로 인해 통신 사이의 물체로 인해 손상되지 않도록 근거리로 셀을 구축해야 한다. 따라서 5G 의 안정적인 통신이 가능하기 위해서는 많은 5G 통신 인프라를 구축해야 하며 이는 경제적 비효율성과 지리적 한계로 인한 음영지역을 발생시킨다.

고속 열차에서 5G 기반의 VR/AR 의 컨텐츠를 사용자들이 경험하기 위해서는 새로운 통신 인프라가 필요하다. 현재 고속 열차의 통신을 담당하고 있는 Rail-side Units(RSUs)는 5G 서비스를 지원하기에 충분하지 않은 숫자이다 [1]. 충분한 수의 RSUs 가 갖추어진다고 하여도 고속으로 이동하는 열차에서 잦은 Handover 가 발생하게 되고 그에 따른 기기 과부하 및 통신 지연이 예측된다. 따라서 본 연구에서는 고속 열차에서의 5G 통신을 위해 RSUs 를 대신하여 UAV-Base Station 을 사용하는 것을 제안한다.

Unmanned Aerial Vehicle(UAV) 기반의 통신은 동적이고 예측 불가능한 서비스 수요를 충족시키는 동시에 지상 기반 기지국의 대안으로 비용을 절감할 수 있는 뛰어난 유연성을 제공할 수 있다 [2]. 현재 UAV 는 이동형 기지국으로서의 활용, 캐싱 장치로의 활용 등 단순한 레저 기기, 군용 장비에서 벗어나 통신을 지원할 수 있는 장비로써 사용하는 연구가 활발히 진행 중이다. 또한, UAV 의 통신 연구에서 가장 크게 문제가 되었던 비행시간, 즉 배터리 소모에 대한 이슈는 수소 배터리 기반의 UAV 가 상용화가 이루어지며 점차 해당 문제가 줄어들고 있다. UAV 의 유연한 이동과 저렴한 비용 등 많은 장점이 존재하지만, 여전히 효율적인 움직임, 배치, 보안 등에서 해결해야 할 문제들이 존재한다. UAV 보다 빠른 고속 열차와 통신을 안정적으로 유지하기 위해서는 고도, 속도, 방향 들을 최적으로 결정하여 경로를 이동하는 것이 중요하다. 따라서 본 연구에서는 고속 열차와 UAV 의 안정적인 통신을 위해 UAV 의 경로 최적화 문제를 해결하고자 한다.

본 논문의 2 장에서는 UAV 통신에서 경로 최적화 문제를 다룬 관련 연구들을 살펴보고, 3 장에서는 본 논문의 실험 환경 구축을 위한 통신 모델과 강화 학습 요소를 설명한다. 4 장에서는 제안한 시스템의 성능 결과에 대한 평가를 다루며, 마지막 5 장은 논문의 결론 및 향후 연구방향을 제시한다.

2. 관련연구

2.1 고정된 사용자에 대한 UAV 경로 최적화 [3]

논문 [3]은 고정된 사용자들에 대해 UAV 의 경로 최적화를 실시하여 평균 통신량을 최대화시켰다. 경로 최적화 기법으로는 Deep Deterministic Poliy Gradient(DDPG) 라는 강화학습을 제안하였고 결과적으로 분산된 사용자들의 통신량이 전체적으로 높아질 수 있었다. 하지만 논문 [3]에서는 위치가 고정된 사용자들에 대해 경로 최적화를 실시했기 때문에 실제 움직이는 차량, 열차와의 통신에서는 한계가 존재한다. 이에 본 논문은 고속으로 움직이는 열차와 UAV 통신 상황에서 UAV 경로 최적화 방안을 제시한다.

2.2 움직이는 사용자에 대한 UAV 경로 최적화 [4]

논문 [4]는 도로 위의 차량과의 UAV 통신에서 UAV 경로 최적화와 자원 할당을 시행하였다. 경로 최적화와 자원 할당을 동시에 최적화하기 위해 수치적인 방법인 Successive Convex Optimization(SCO)라는 기법을 사용하였다. 하지만 논문 [4]에서의 UAV 는 고도가 고정되어 있어 통신 범위가 달라지는 것에 대해 고려는 하지 않았다. 본 논문은 UAV 의 고도 또한 경로 선정의 변수로 고려하여 최적의 통신 범위를 가질 수 있도록 하였다.

3. 제안사항

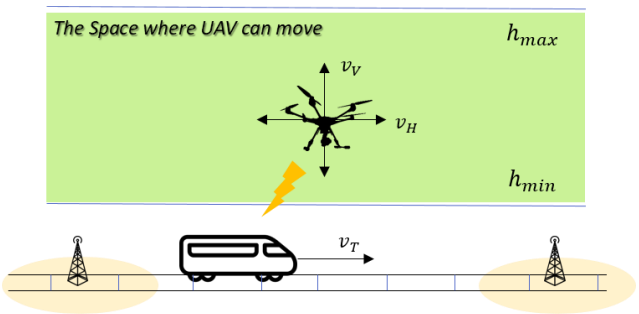


그림 1. 시스템 모델

본 논문은 그림 1 과 같이 RSUs 가 지원하지 못하는 통신 범위에 대해 UAV 를 이용하여 통신을 지원할 수 있도록 한다. 고속 열차는 v_T의 속도를 가지며 UAV 는 v_V의 수직 속도와 v_H의 수평 속도를 가지게 된다. 또한 UAV 는 고도의 제한이 있으며 h_{min}과 h_{max} 사이의 고도에서만 이동이 가능하다. 본 논문은 위와 같은 환경에서 UAV 가 열차의 속도에 따라 최적의 v_H, v_V를 결정하여 최적의 경로 이동을 할 수 있는 방법을 제안한다.

$$\mathrm{R}(h_t) = h_t \cdot \tan\frac{\omega}{2} \tag{1}$$

식 (1)은 높이 h_t와 통신이 가능한 각도 ω에 따른 통신 범위의 반경을 도출하기 위한 식이다 [5]. 통신 대상인 열차의 속도가 빠르다면 UAV 는 고도를 높여 통신 범위를 늘리는 것이 지속적인 통신에 유리할 것이다. 그렇지 않을 때는 낮은 고도에서 높은 통신량을 보장하는 것이 유리하다. 이에 본 논문은 강화학습을 적용하여 열차의 속도에 따른 UAV 의 최적 경로를 빠르고 정확하게 찾을 수 있도록 제안한다. 가상 환경 구현을 위해 3.1 절의 통신 모델과 3.2 절의 강화학습 환경을 사용한다.

3.1 통신모델

Path Loss Model: UAV 와 열차 간의 통신 과정에서 거리(d)에 따른 신호 감쇄 효과로 인한 SNR(신호 대 잡음 비)을 계산하기 위해 Air-To-Ground Pass Loss 모델을 사용했다 [6]. 식 (2)은 거리(d), 빛의 속도(c) 와 UAV 의 통신 주파수(f_0)을 통해 UAV 와 열차 간의 통신 손실을 구하는 식이다. $P(LoS)$ 는 통신 간에 장애물에 없는 line-of-sight 가 될 확률이고 $P(NLoS)$는 통신 간에 장애물이 있을 확률이다. 따라서 $P(LoS) = 1 - P(NLoS)$라는 식을 얻을 수 있다. 그리고 η_{LoS}와 η_{NLoS}는 각각, LoS 와 $NLoS$ 에서 추가적인 감쇄 수치이다. 식 (6)의 a와 b는 설정 환경에 따라 변하는 상수이다.

$$PL(dB) = 20\,log_{10}(\frac{4\pi d f_0}{c}) + P(LoS)\eta_{LoS} + P(NLoS)\eta_{NLoS} \tag{2}$$

$$P(LoS) = \frac{1}{1 + a\exp(-b(\theta - a))} \tag{3}$$

이렇게 구한 통신 손실(PL)을 식 (4)와 식 (5)에 차례로 대입하여 통신 세기(P_r)와 SNR을 각각 구할 수 있다.

$$P_r(d) = P_t + G_t - PL + G_r - L \tag{4}$$

$$SNR = \frac{Signal\ Power}{Noise\ Power} = \frac{P_r(d)}{N_0} \tag{5}$$

Data Rate Model: Shannon-Hartley 이론에 기반하여 UAV 와 열차 간의 통신량에 대한 식 (6)을 구할 수 있다. 따라서 대역폭(B)에 대해 앞서 구한 SNR 을 식 (6)에 대입하여 통신량(D)을 도출할 수 있다.

$$D_t = B\,log_2(1 + SNR) \tag{6}$$

3.2 강화학습 환경

강화학습을 위해 Unity 의 ML-Agents 를 사용하였다 [7]. 해당 라이브러리는 다양한 강화학습 환경을 구축하는데 유용한 라이브러리로써 본 논문의 학습 환경 구축에 사용되었다. 강화학습은 Tensorflow 을 사용하였으며 Soft-Actor-Critic 이라는 방법을 사용하여 학습하였다. 환경은 그림 1 과 같이 하나의 고속열차와 한대의 UAV 와의 통신에서 다루었다. 열차는 직선의 선로를 이동한다고 가정하였고 최저 속도($70\ \mathrm{m/s}$)에서 최대 속도($90\ \mathrm{m/s}$) 사이의 속도로 이동한다. UAV 는 허용 고도($100\mathrm{m} \sim 200\mathrm{m}$)에서 최대 $25\mathrm{m/s}$의 수직 속도 v_V와 최대 $60\mathrm{m/s}$의 수평 속도 v_H을 가진다. 하나의 에피소드

T는 일정한 시간 간격으로 나누어 진다.

$$\mathrm{T} = \{t_0, t_1, t_2, \cdots, t_i, \cdots\} \tag{7}$$

State: 시간 t_i에서 상태 s_i는 식 8 과 같이 정의한다.

$$\mathrm{s}_i = \{p_u, p_t, v_T\} \tag{8}$$

p_u와 p_t는 각각 UAV 의 위치와 열차의 위치이며 v_T는 열차의 속도이다.

Action: 시간 t_i에서 행동 a_i는 식 9 과 같이 정의한다.

$$\mathrm{a}_i = \{v_V, v_H\} \tag{9}$$

v_V와 v_H는 방향의 가진 속도로써 UAV 의 방향과 속력을 결정할 수 있다. 학습 초반에는 행동은 최대 속도를 벗어나지 않는 선에서 무작위로 실행되며 후반으로 갈수록 보상이 커지는 행동을 하게 된다.

Reward: 시간 t_i에서 보상 r_i는 식 10 과 같이 정의한다.

$$\mathrm{r}_i = \begin{cases} 0.1 \cdot \dfrac{\mathcal{D}}{i} & out\ coverge \\ -1 & out\ altitude \\ D_t & else. \end{cases} \tag{10}$$

$$\mathcal{D} = \sum_{t=0}^{i} D_t \tag{11}$$

보상은 3 가지의 경우로 나뉘게 된다. 먼저 열차가 UAV 의 통신 범위를 벗어나게 되면 에피소드를 종료하며 $0.1 \cdot \frac{\mathcal{D}}{i}$을 보상하게 된다. 식 11 의 D_t는 해당 시간의 통신량을 의미하며 결과적으로 에피소드 동안의 평균 통신량을 보상한다. 다음 경우는 UAV 가 허용 고도를 넘어 이동했을 때는 에피소드를 종료하며 -1의 보상을 주어 해당 행동이 나오지 않게끔 유도한다. 그 외의 통신이 안정적으로 이루어지는 경우는 해당 시간에 대한 처리량 D_t을 보상하며 에피소드를 그대로 진행한다.

4. 실험결과

표 1. 변수 정의 및 사용 값

Notion	Description	Value
ω	*The Angle of the sensing cone*	90°
f_0	*Transmitter frequency*	$5\ GHz$
P_t	*Transmitting power*	$10\ dBW$
$G_t,\ G_r$	*Antenna gains for the transmitter and receiver*	$8\ dB, 0\ dB$
L	*Total System Losses*	$8\ dB$
B	*Bandwidth*	$20\ MHz$
N	*Noise Power*	$107\ dBW$
a, b	*Constants on the environment*	$4.88, 0.49$
η_{LoS}, η_{NLoS}	*Additional losses in LoS and NLoS*	$0.1, 21$

표 2. 강화학습 서버 주요 사양

Part	Spec
CPU	*Intel® Core™ i5-8500 @ 3.00GHz*
GPU	*GeForce GTX 1660 Ti*
RAM	*16.0GB*

표 1 은 실험에 사용된 변수들의 정의와 수치를 나타낸다. UAV 가 지원하는 통신을 5G 서비스라고 가정하였기 때문에 주파수와 대역폭을 표 1 에서와 같이 정의하였다. 또한, 교외 상황에서의 통신이라고 가정하여 그에 따른 환경 변수(a, b), 추가 감쇄 수치$(\eta_{LoS}, \eta_{NLoS})$가 정의되었다. 표 2 에서는 강화학습을 실행한 서버의 사양을 확인할 수 있다.

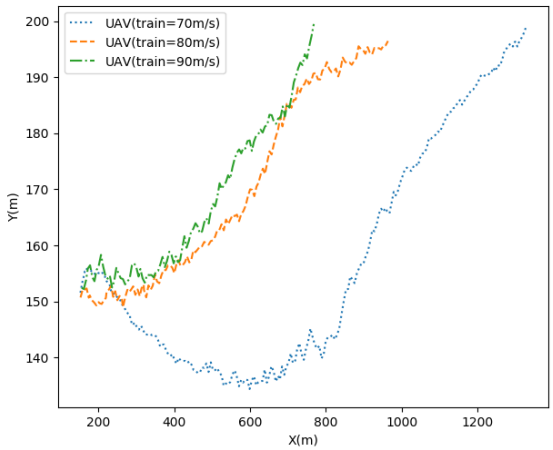


그림 2. 열차 속도에 따른 UAV 경로

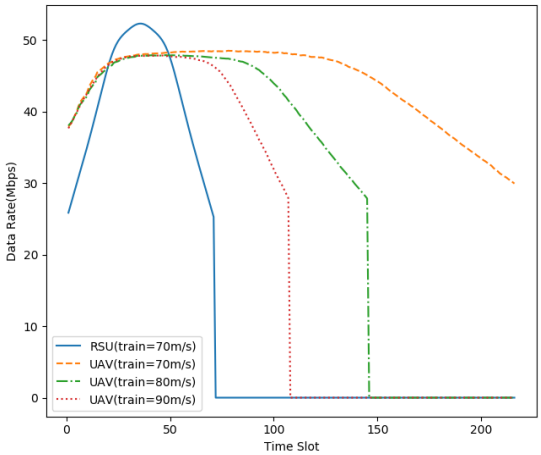


그림 3. 열차 속도에 따른 통신량

그림 2 는 열차의 속도에 따른 UAV 경로를 시간 순으로 나타낸 것이다. 열차의 속도가 빨라짐에 따라 UAV 가 통신을 지원할 수 있는 시간이 줄어드는 것을 알 수 있다. UAV 는 공통적으로 빠른 열차와의 통신을 안정적으로 유지하기 위해 고도를 높이는 것을 확인할 수 있다. 상대적으로 열차가 느린 경우 즉 그림 2 에서 열차 속도가 70m/s 일 때 UAV 는 고도를 낮추어 통신량을 높이려는 움직임 또한 관찰할 수 있다.

그림 3 은 열차의 속도에 따른 통신량을 시간 간격으로 나타낸 것이다. 그림 3 을 통해 일반적인 지상 RSUs 와 UAV 를 비교하면 더욱 오래 통신 서비스를 지원할 수 있다는 것을 알 수 있다. 따라서 UAV 통신은 통신 지원 범위가 넓힐 수 있고 따라서 Handover 를 줄여 기기 과부화 또는 지연 시간을 줄일 수 있을 것으로 기대된다. 그리고 열차의 속도가 빨라짐에 따라 UAV 는 통신을 지속하기 위해 더욱 높은 고도를 유지해야 한다. 이것으로 인해 결과적으로 평균 통신량은 줄어드는 것을 알 수 있다.

5. 결론 및 향후 연구

본 논문에서는 고속 열차와 UAV 와의 통신 상황에서 고속 열차에 속도에 따른 UAV 의 경로 최적화 방안에 관해 연구하였다. 최적의 경로를 찾기 위해 강화학습을 Unity ML-Agents 환경에서 사용하였다. 결과적으로 제안한 방법을 통해 기존의 RSUs 와 통신을 할 때보다 UAV 가 열차에 따라 이동을 하면서 통신 지원 범위를 넓일 수 있어 효율이 좋았다. 또한, 열차의 속도에 따라 UAV 가 고도를 조정하며 최대한 길고 안정적으로 통신을 지원할 수 있었다. 앞으로 본 논문에서 다루지 않았던 다수의 UAV 통신을 통해 통신 지원 범위를 더욱 넓히는 방안과 배터리 효율 및 충전에 대한 변수까지 고려하여 본 논문을 발전시킬 것이다.

ACKNOWLEDGMENT

이 논문은 2019 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-01287, 분산 엣지를 위한 진화형 딥러닝 모델생성 플랫폼)
*Dr. CS Hong is the corresponding author.

참 고 문 헌

[1] Zhou, Yuzhe. "Future communication model for high-speed railway based on unmanned aerial vehicles." arXiv preprint arXiv:1411.3450 (2014).

[2] Yu Min Park, Min Kyung Lee, and Choong Seon Hong. "Multi-UAVs Collaboration System based on Machine Learning for Throughput Maximization." 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS). IEEE, 2019.

[3] Qi, Hang, et al. "Energy Efficient 3-D UAV Control for Persistent Communication Service and Fairness: A Deep Reinforcement Learning Approach." IEEE Access 8 (2020): 53172-53184.

[4] Samir, Moataz, et al. "Joint Optimization of UAV Trajectory and Radio Resource Allocation for Drive-Thru Vehicular Networks." 2019 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2019.

[5] Trotta, Angelo, et al. "Joint coverage, connectivity, and charging strategies for distributed UAV networks." IEEE Transactions on Robotics 34.4 (2018): 883-900.

[6] Al-Hourani, Akram, Sithamparanathan Kandeepan, and Simon Lardner. "Optimal LAP altitude for maximum coverage." IEEE Wireless Communications Letters 3.6 (2014): 569-572.

[7] Juliani, Arthur, et al. "Unity: A general platform for intelligent agents." arXiv preprint arXiv:1809.02627 (2018).

비트코인 메모리풀의 자카드 지수 기반 유사도 차이 원인 분석

*김대용, 주홍택
계명대학교*, 계명대학교

*imdy1207@gmail.com, juht@kmu.ac.kr

Analysis of Causes of Similarity Difference based on Jaccard Index for Bitcoin Memory Pool

*Kim Dae Yong, Ju Hongtaek
Keimyung Univ.*, Keimyung Univ.

요 약

비트코인 네트워크에서 메모리풀은 블록에 포함되기 전의 트랜잭션을 보관하며 각 노드는 연결된 피어들이 다르고 네트워크 연결 상태가 상이하는 등 다양한 원인으로 인해 메모리풀에 보관하고 있는 트랜잭션의 차이가 발생한다. 본 논문에서는 실행 조건이 동일한 4 개의 비트코인 노드에서 메모리풀을 관찰하여 메모리풀의 유사도를 자카드 지수 기반으로 분석하고 유사도 차이의 원인을 분석하여 그 결과를 제시한다.

Ⅰ. 서 론

비트코인에서 메모리풀은 트랜잭션이 검증된 후 블록체인에 반영되기 전에 보관되는 공간으로 모든 트랜잭션들이 필수적으로 거치는 저장 공간이다. 비트코인 네트워크에 참여하는 노드들의 상태, 트랜잭션과 블록의 전파 그리고 노드의 메모리풀 관리에 따라서 각 노드의 메모리풀에 저장된 트랜잭션은 차이가 발생한다. 이러한 차이는 노드의 마이닝 성능에 영향을 미치고 비트코인 전체 네트워크 관점에서도 불필요한 메시지를 발생시키는 등 블록체인의 전체 성능과도 밀접한 관련이 있다.

우리는 이전 논문에서 비트코인과 이더리움의 메모리풀 유사도를 분석하였다[1]. 해당 논문에서는 동일한 조건의 4 개 노드에서 메모리풀의 유사도 분석을 실시하였으나 유사도 차이의 원인을 제시하지 못하였다. 본 논문에서는 이전 논문의 결과를 바탕으로 메모리풀 트랜잭션의 유사도를 각 노드의 상대적 관점에서 자카드 지수를 좀더 면밀히 분석하였고 분석의 결과로 도출된 유사도의 차이점이 발생하게 된 원인을 규명하였다.

II. 관련 연구

2.1 비트코인 블록 전달

비트코인의 메모리풀은 비트코인 네트워크에서 검증은 되었지만, 승인되어 채굴자들에 의해서 생성된 블록에 아직 포함되지 않은 트랜잭션들이 저장되어 대기하는 영역으로, 그 크기는 노드의 RAM 용량에 따라 결정된다[2]. 노드가 네트워크로부터 생성된 블록을 전파 받으면, 해당 블록에 포함된 모든 트랜잭션은 메모리풀에서 제거된다. 즉 트랜잭션이 블록에 포함되면 메모리풀에서 제거된다. 또한 메모리풀의 크기가 설정된 임계값 보다 커지게 되면 수수료가 낮은 트랜잭션부터 제거된다[3]. 블록 생성 시에도 수수료가 높은 트랜잭션 순으로 우선적으로 채택이 되므로 메모리풀의 모니터링을 통해 해당 시점의 합리적인 수수료 예측도 가능 하다.[4]

신규 블록 데이터 릴레이(Block Relay)[5]는 비트코인 네트워크에서 신규 블록이 생성되어, 해당 블록이 모든 노드로 전파하는 과정에서 각 노드의 블록 전달 방법을 말한다. 블록 데이터 릴레이는 최근 압축 블록 전달 방식(Compact Block Relay)이 도입되어 상당한 성능 향상이 이루어졌다[6]. 이 방식은 기존에 블록을 전달할 때 블록 헤더와 블록에 포함된 모든 트랜잭션을 함께 전달하는 방식에서 블록 헤더만 먼저 전달하고 이 헤더에 포함된 트랜잭션은 전달받은 노드의 메모리 풀의 트랜잭션을 활용하여 성능을 향상시켰다.

그림 1 에서 보이는 바와 같이 압축 블록 전달 방식은 저 대역폭 릴레이(Low Bandwidth Relay) 방법과 고 대역폭 릴레이(High Bandwidth Relay) 방법이 선택적으로 사용된다. 고 대역폭 릴레이 방법은 전달받을 노드가 이 블록을 가지고 있지 않다고 가정하여 INV 메시지와 GETDATA 메시지 교환을 삭제하여 성능을 향상시켰다. 고대역폭 릴레이 방식은 2016 년 8 월부터 적용되었으며, 이러한 전달방식의 장점은 전파할 블록의 트랜잭션을 모두 보낼 필요가 없으므로 데이터 전송량을 줄여, 네트워크의 트래픽을 감소시킬 수 있다. 즉, 전달하는 노드와 전파 받는 노드간 메모리풀 트랜잭션들의 유사도가 높을수록 트래픽이 상당히 감소한다고 볼 수 있다. 따라서 본 논문에서는 이러한 과정에서 필요한 지표인 메모리풀 트랜잭션의 유사도를 측정하여 분석해 보았다.

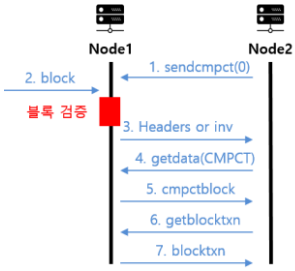


그림 1. 저대역폭 릴레이

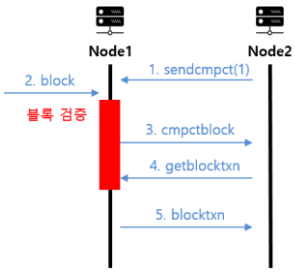


그림 2. 고대역폭 릴레이

.

2.2 메모리 풀 유사도 분석 연구

‘ 비트코인 노드 메모리 풀 유사도 분석’ [7]에서는 비트코인 메모리풀의 트랜잭션 유사도를 비교하여 노드들이 지역적으로 유사한 위치에 있더라도 메모리풀 의 유사도가 높지 않는다는 사실을 보였다. 이 논문은 지역적인 특성 뿐만 아니라 네트워크, 하드웨어 성능 및 조건들을 동일 하게 구성하여 연구를 진행하였으나 유사도 차이에 대한 상세 분석이나 차이가 나는 원인을 제시하지 못하였다.

우리가 이전에 작성한 논문에서[1] 우리는 비트코인과 이더리움 두 가지 플랫폼의 메모리풀 유사도를 측정하였다. 하지만 해당 논문에서도 메모리풀에서 발생하는 트랜잭션의 변화에 대한 구체적인 원인을 제시하지 못하였다. 본 논문에서는 유사도 뿐만 아니라 각 노드를 기준으로 유사도를 측정하였고, 해당 데이터를 통해, 트랜잭션이 변화하는 원인을 설명하였다.

Muhammad Saad 는 메모리풀을 공격 표적으로 하는 새로운 형태의 공격 방법을 제시하였고, 그러한 공격에 의해 합법적인 거래를 하는 사용자들이 지불하는 수수료에 미치는 영향에 대해 연구하였다[8]. 이러한 공격을 억제하기위해 메모리풀의 크기를 최적화하고 공격을 예방하기 위한 수수료 기반으로 방어 대응책을 설계하였다. 또한 시뮬레이션을 통해 이 논문에서 제시된 방어 모델을 평가하고 다양한 조건의 공격을 통해 그 유용성을 검증하였다. 이 논문에서 메모리풀의 크기와 수수료의 데이터를 수집하였는데, 단일 노드에서 시간당 트랜잭션의 개수만을 측정하였다. 우리는 위의 분석과정을 참고로 하여 4 개의 노드로부터 데이터의 변화를 관찰하고 데이터를 분석하였다.

Imtiaz 는 비트코인 네트 워크에서 노드들이 추가되고 제거되는 빈번한 변화 현상에 대해 기술하였다[9]. 이 현상 규명을 통해 앞에서 설명한 신규 블록 데이터 릴레이간 발생하는 압축 블록 전달 과정에서 지연이 발생하여 성능이 감소 한다는 사실을 밝혔고, 이 현상이 메모리 풀 트랜잭션들의 차이를 발생시키며, 메모리풀의 동기화를 통해 이러한 현상을 완화 할 수 있음을 제시하였다. 해당 논문을 통해 메모리풀 트랜잭션의 유사도는 블록체인을 모니터링 하기위한 유용한 지표로써 사용 될 수 있음을 알 수 있다.

III. 실험 환경 및 데이터 수집

본 논문에서 메모리풀의 유사도를 분석하고 결과를 통해 메모리 풀에서 발생한 변화들을 해석하는 과정은 기초적인 데이 터를 수집하고, 수집된 데이터를 분석하는 과정으로 진행 하였다.

3.1 실험 환경

비트코인 노드의 메모리풀에 있는 트랜잭션 정보를 수집하기 위해서는 가장 대중적으로 사용되는 비트코인 코어 프로그램을 설치하여 활용하였다[10]. 설치한 비트코인 코어 프로그램은 풀 노드(Full Node)로 동작하며 RPC 접속을 통해 노드로부터 메모리 풀의 데이터 뿐만 아니라 분석에 필요한 데이터를 수집하였다. 풀 노드는 비크코인의 모든 기능을 수행하며 약 300GB 이상의 충분한 저장소와 100MB/s 이상의 대역폭이 요구된다.

하나의 노드에서 최대로 연결가능한 피어의 개수는 8 개이고, 네트워크에 참여된 노드들의 상태에 따라 연결 상태 또한 변하게된다.[11] 때문에 수집하는 모든 노드가 서로를 피어로 연결하여 외부의 노드와 연결하지 않는 상태를 방지하기 위해, 최대로 연결 가능한 피어의 개수의 절반인 4 개의 노드를 운용하여 데이터 분석을 진행하였다. 또한 동일한 환경에서 데이터를 수집하기 위해 동일한 성능의 'Dell PowerEdge R610 Server (Intel(R) Xeon(R) CPU X5650 @ 2.67GHz, 8G RAM)'를 물리적으로 독립된 환경과 동일 IP 대역을 가진 네트워크에 구성하여 운용하였으며, 비트코인 노드 설치 및 동기화를 동시에 진행하였다. 또한 노드의 동기화가 완료된 후 데이터 수집을 시작하였으며, 수집 당시 설치된 모든 노드의 비트코인 코어의 버전은 가장 많은 노드들이 사용중인 0.18.0 을 사용했다.

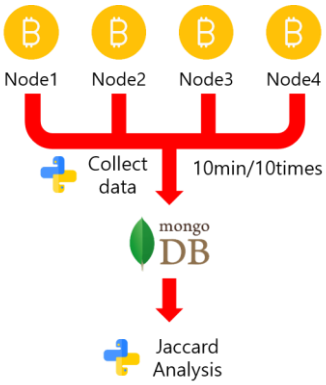


그림 3. 데이터 수집 및 저장 과정

메모리 풀에 저장된 트랜잭션 데이터를 수집하는 과정을 그림 2 가 보여 주고 있다. 수집 데이터들은 비트코인 코어에서 제공하는 RPC 명령어를 사용해 4 개의 노드에서 동시에 수집된다. ‘ getrawmempool’ RPC 명령어를 사용하면 각 노드의 메모리풀에 존재하는 트랜잭션의 목록을 JSON 배열 형태로 출력한다. 이 데이터는 10 분간격으로 총 10 회 수집되며 MongoDB 에 저장된다. 저장된 데이터를 기반으로 메모리풀 트랜잭션 자카드 지수 및 트랜잭션의 변동을 분석한다.

수집한 데이터는 메모리풀에 있는 트랜잭션들의 해시값이다. 트랜잭션 해시값은 트랜잭션을 대표하는 고유의 값으로 메모리풀에 존재하는 트랜잭션들의 개수를 파악할 수 있고, 또한 해시값을 통해 각 노드의 메모리 풀에 저장된 트랜잭션을 비교 분석할 수 있다. 해당 데이터들은 2019 년 10 월 31 일 오후 2 시 45 분부터 10 분 간격으로 10 회 비트코인 코어 RPC 명령어를 사용하여 수집하였다. 수집된 모든 데이터의 크기는 총 7.38MB 이며, 각 수집 과정에서 실시간으로 데이터를 수집 하였다.

IV. 자카드 유사도 분석

본 논문에서 측정한 메모리풀의 유사도는 측정 당시 메모리풀에 존재하는 트랜잭션들의 고유한 해시값을 바탕으로 자카드 지수와 자카드 거리를 산출하여 분석하였다. 또한 유사도 분석과 메모리풀의 트랜잭션 변동값을 분석하여 메모리풀의 유사도 차이가 발생하는 원인을 밝힌다.

4.1 자카드 유사도 차이

그림 3 은 자카드 지수와 자카드 거리를 측정하는 과정에서 필요한, 4 개의 노드에 있는 메모리풀의 교집합과 합집합을 보여 주고 있고 수식 (1)과 수식 (2)는 자카드 지수와 거리의 정의이다. 자카드 지수[12]는 여러 집합들

사이의 유사도를 측정하는 방법으로, 0~1 사이의 결과값을 가지게 교집합의 원소의 개수를 합집합의 원소의 개수로 나눈 값이다. 측정된 자카드 지수의 값이 1 에 근접할수록 유사하다고 볼 수 있다. 자카드 거리는 여러 집합들 사이의 비유사도를 측정하는 측도이며, 합집합의 원소의 개수 에서 교집합 원소의 개수를 뺀 값을 합집합 원소의 개수로 나눈 값으로, 즉 1 에서 자카드 지수를 뺀 값이다. 측정된 자카드 거리의 값이 0 에 근접할수록 유사하다고 볼 수 있다. 본 논문에서 각 노드의 메모리풀은 트랜잭션을 원소로 하는 집합으로 볼 수 있다.

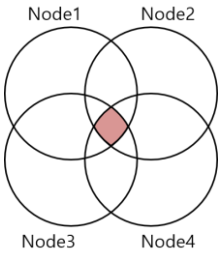


그림 4. 노드의 교집합

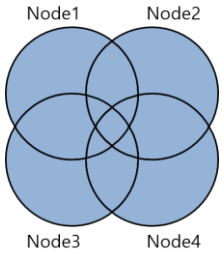


그림 5. 노드의 합집합

$$J(A, B, C, D) = \frac{|A \cap B \cap C \cap D|}{|A \cup B \cup C \cup D|}$$

수식 1. 자카드 지수

$$d_{jacard}(A, B, C, D) = \frac{|A \cup B \cup C \cup D| - |A \cap B \cap C \cap D|}{|A \cup B \cup C \cup D|}$$

수식 2. 자카드 거리

수집된 데이터를 바탕으로 측정한 모든 노드의 메모리풀 트랜잭션의 자카드 지수를 통해 분석한 결과는 '그림 4'과 같다. 8 번째 데이터를 제외한 모든 결과값이 거의 1 에 근접했다. 즉, 8 번째 데이터를 제외한 나머지 모든 측정 당시의 노드들의 메모리풀 유사도가 높다는 것을 확인할 수 있었다.

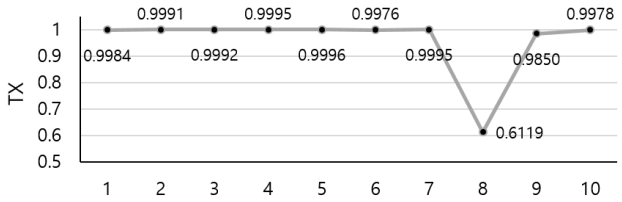


그림 6. 전체 노드의 자카드 유사도

8 번째 데이터의 유사도 낮다는 것은 모든 노드가 공통적으로 가지고 있는 트랜잭션이 적다는 의미이며 이는 특정 노드에서 공통적으로 가지고 있는 트랜잭션이 다를 수도 있고 노드들이 비슷하게 다른 트랜잭션을 가질 수도 있다고 해석할 수 있다. 따라서 앞에서 나타난 8 번째 데이터에서 유사도가 감소하는 현상이 어떤 노드를 기준으로 발생했는지를 파악하기 위해 각 노드들을 기준으로 나머지 노드들과 자카드 거리를 분석하였고, 그 결과는 '그림 5'와 같다. 그림 5 의 좌측 상단에 있는 그래프는 노드 1 을 기준으로 측정한 각 노드들과 비교한 자카드 거리로 노드 1 과 다른 노드들 간의 비유사한 정도를 나타낸다.

모든 그래프의 8 번째 데이터에서 노드 1, 2, 4 간의 자카드 거리는 0 에 근사한 값이 나타났지만, 노드 1,2,4 와 노드 3 과 비교한 자카드 거리는 약 0.4 에 가까운 값이 나타났다. 이러한 수치를 바탕으로 노드 3 에서의 메모리풀의 트랜잭션이 다른 3 개의 노드(노드 1, 2, 4)의 트랜잭션과 차이가 있음을 알 수 있다. 또한 노드 1, 2, 4 간 자카드 거리의 측정값들이 0 에 근사한 것을 통해 노드 1, 2, 4 의 메모리풀은 비슷한 트랜잭션을 보유하고 있음을 알 수 있다.

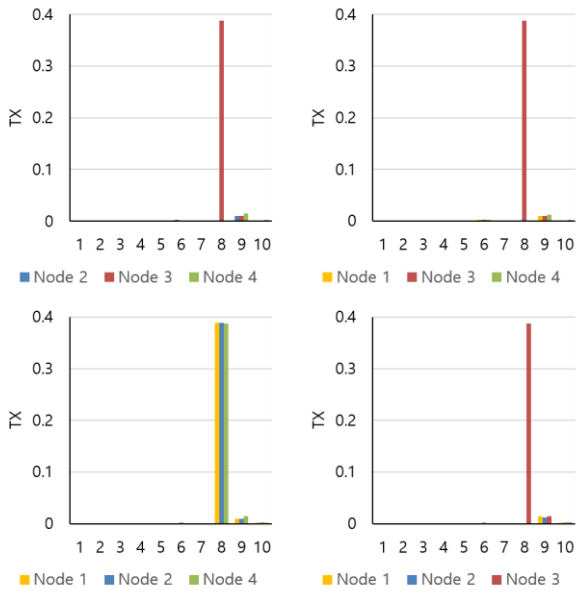


그림 7. 각 노드의 자카드 거리

4.2 유사도 차이 원인 분석

메모리풀에서 트랜잭션의 변화가 발생하는 경우 '표 1'와 같이 5 가지 경우로 나누어진다.

변동 분류	변동 원인
트랜잭션이 증가하는 경우	① 신규 트랜잭션이 발생하여 노드로 전파 받은 경우
	② 제거된 블록의 트랜잭션들이 다시 메모리풀로 돌아온 경우
트랜잭션이 감소하는 경우	③ 신규 블록이 생성된 경우
	④ 트랜잭션의 수수료값이 기준값 보다 낮은 경우
	⑤ 메모리풀의 메모리가 가득찬 경우

표 1. 메모리풀 트랜잭션 변동 원인

'표 1'의 '①'은 실시간으로 트랜잭션이 유입 되어 메모리풀에 트랜잭션이 저장되는 경우이다. 이러한 상황은 비트코인 네트워크에서 트랜잭션을 실시간으로 받아 발생하는 경우이므로, 주기가 규칙적이거나 전파 받는 트랜잭션의 수가 일정하지 않다[13]. 따라서 메모리풀의 트랜잭션이 불규칙적으로 증가하기 때문에 몇 개가 메모리풀에 유입되었는지를 정확하게 예측하기가 어렵다. '②'는 블록에 포함된 트랜잭션들이 고아 블록(Orphan Block)으로 판명되어 그 블록에 포함되었던 트랜잭션이 다시 메모리 풀로 돌아오는 경우이다. 고아 블록은 노드에서 자체적으로 판단하며, 블록체인 모니터링 웹 페이지에서 제공하는 데이터를 기준으로 측정 당일의 평균 블록당 트랜잭션의 수가 2,226 개라는 점을 감안하면 이 경우는 한번에 대량의 트랜잭션이 유입되기 때문에 '①'의 경우보다 더 많은 수의 트랜잭션이 메모리풀에 추가된다[14]. '③'의 경우 앞서 설명했듯이 데이터 측정 날짜 기준 하나의 블록에는 평균적으로 약 2,226 개 정도의 트랜잭션이 포함되어 있다. 즉, 블록을 전파 받았을 경우 약 2,226 개의 트랜잭션이 메모리풀에서 제거되고 있다[14]. 따라서 메모리풀에서 트랜잭션의 수가 급격하게 감소하는 경우라고 볼 수 있다. '④'의 경우는 감소하는 트랜잭션의 수와 주기가 불규칙적으로 발생하므로 예측하기 어렵고, '⑤'의 경우는 측정된 값들 중 트랜잭션의 수가 가장 큰 임계점 이상일 경우 발생할 것으로 예측가능하다.

트랜잭션이 증가하는 경우와 감소하는 경우를 고려하면 증가하고 감소하는 원인을 메모리 풀의 트랜잭션 수로 예측할 있음을 알 수 있다. 메모리 풀의 트랜잭션 수가 증가 할 때 '①'의 경우는 비트코인에서 데이터 측정 간격인 10 분당 발생하는 트랜잭션의 수를 예측하기 어렵고, '②'의 경우는 블록당 포함된 평균 트랜잭션의 수가 데이터 측정일 기준으로 2,336 이고, 최근 1 년 평균값 또한 2,196 개를 유지함으로, 블록당 트랜잭션의 개수의 변동이 크지 않다. 트랜잭션의 수가 감소할 때 ③은 10 분마다 주기적으로 약 2,000 여개의 값이 발생하고 ④의 경우는 수의 변동이 크고 불규칙적으로 발생한다. ⑤의 경우는 빈번하게 발생하지 않음을 알 수 있다.

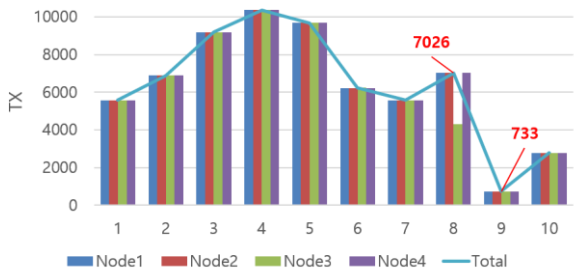


그림 8. 노드별 트랜잭션 수 그래프

'그림 6'은 각 노드 당 트랜잭션들의 수와 모든 노드에서의 전체 트랜잭션의 수를 나타낸 그래프이다. 1~4 번째 측정까지는 데이터가 연속적으로 증가 하는 추이를 보였고, 4 번째 측정에서 트랜잭션의 값이 10,361 개로 가장 높게 나타났다. 4~7 번째 측정된 데이터는 감소하는 추이를 보였지만 4~5, 6~7 에서는 감소한 값은 각각 678, 654 개로 5~6 의 3,451 개에 비해 비교적 작은 값을 보였다. 8 번째 에서는 노드 3 을 제외한 모든 노드에서 트랜잭션이 증가했다. 9 번째 데이터에서는 733 개로 측정된 값들 중 가장 낮은 값이 측정되었고, 이후 10 번째 부터 다시 증가하는 추이를 보이고 있다. 트랜잭션이 증가하는 경우 중 1→2, 2→3, 7→8 구간에서는 1,171~1,448 사이의 값으로 비슷한 추이를 보이고, 2→3, 9→10 에서는 각각 2,304 과 2,043 으로 급격한 증가를 보인다. 그리고 감소하는 경우에는 4→5 와 6→7 구간에서는 각각 678, 654 개로 5→6 과 8→9 구간의 3,451 과 6,293 개에 비해 소폭 감소하는 경우를 보였다.

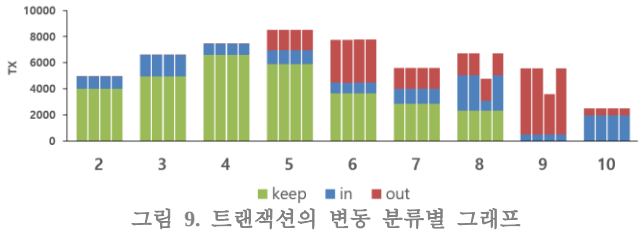


그림 9. 트랜잭션의 변동 분류별 그래프

'그림 6'의 그래프에서 메모리 풀의 변화 요인을 정확히 분석할 수 없어 '그림 7'과 같이 데이터의 수집간 이전 수집 데이터와 비교하여 유지된 트랜잭션의 수(keep)와 추가된 트랜잭션의 수(in), 감소한 트랜잭션의 수(out) 를 계산하였다. 2~4 의 구간에서 감소한 것은 감소한 트랜잭션의 수가 블록에 포함된 평균 트랜잭션의 개수(2,336)에 비해 훨씬 작은 값을 가지고 있고, 비트코인 코어를 통해 트랜잭션 정보를 확인한 결과 유효하지 않은(블록에 포함되지 않은) 트랜잭션으로 밝혀졌다. 때문에 '표 5'에서 '④'혹은 '⑤'의 경우 둘 중에 해당된다고 유추할 수 있는데, '그림 7' 그래프의 2~5 구간에서 지속적으로 증가하고 있는 것으로 보아 메모리풀의 크기가 임계점을 초과하지 않았다는 것을 알 수 있다. 따라서, '④'경우에 해당된다는 것을 알 수 있다.

'그림 7'의 그래프 2, 4, 5, 6, 9 에서 추가된 트랜잭션의 수는 722~1,478 개로 블록에 포함된 평균 트랜잭션의 개수(2,336)에 훨씬 못 미치는 값을 가지고 있으므로, '표 1'의 '①'의 경우에 해당한다고 해석할 수 있다. 5~9 구간의 경우 감소한 트랜잭션의 수는 최소 2,151 에서 최대 7,024 의 값을 가지고 있는 것을 바탕으로 1 개 이상의 블록을 전파 받아 트랜잭션이 감소한 경우임을 예측할 수 있다('표 1'의 '③'). 6, 9 번째 데이터의 경우 감소한 값이 각각 4,565 와 7,018 개 이상으로 나타난 것을 바탕으로 블록을 전파 받는데 지연이 생겨 해당 데이터에서 다수의 블록이 전파되어 감소했다는 것 또한 유추 가능했다. 실제로 비트코인 코어 RPC 를 사용하여 9 번째 데이터에서 감소한 트랜잭션들을 분석한 결과 해당 트랜잭션들이 담긴 블록의 번호들은 '601757', '601758', '601759'번으로 총 3 개의 블록을 전파받아 메모리풀의 트랜잭션이 제거 되었음을 확인할 수 있었다.

본 실험에서는 트랜잭션의 해시값 만을 이용하여 세밀한 메모리풀의 데이터들을 분석하는데 한계가 있었다. 새로 전달된 트랜잭션을 모니터링하거나 고아블록이라고 판단하는 시점 그리고 블록 전파 시점, 메모리 풀 리셋 시점 등을 정확히 알아내고 블록에 저장된 트랜잭션과 비교 분석을 하면 좀더 명확한 원인 규명이 가능하다.

이번 실험은 동일한 환경의 4 개 노드에 대한 유사도 분석으로 실제로 비트코인에서 운용중인 모든 노드들의 데이터와는 차이가 존재한다. 그리고 메모리풀의 정보는 실시간으로 변하므로, 데이터 처리 과정에서 발생하는 지연으로 인해 누락되거나 변경될 가능성 존재한다. 또한 자카드 지수만으로 분석 가능한 정보가 제한적이다. 마지막으로, 현재 메모리풀의 트랜잭션이 증가하고 감소하는 정확한 기준이 제시되지 않았다. 이러한 문제점들을 참고하여 향후 연구에 반영할 계획이다.

Ⅲ. 결론

본 연구에서는 자카드 지수를 통해 트랜잭션의 변동을 관찰하여, 자카드 지수를 통해 메모리풀에서 발생하는 다양한 현상들을 분석하였다. 동일한 환경에서 4 개의 노드가 실행되어 메모리 풀의 거의 유사할 것으로 추측되었으나 자카드 지수 분석으로 대부분의 경우 유사하나 반드시 그렇지 않음을 알게 되었고 그 원인도 분석하였다. 하지만 앞서 언급한 데이터의 한계점으로 인해 좀 더 다양하고 구체적인 분석이 어려웠다. 이러한 문제점들을 개선하고 해결하기 위해 다양한 데이터들을 수집하여 분석할 계획이다.

우선 메모리풀에서 트랜잭션이 추가되고 삭제되는 기준을 명확하게 파악하기 위해서 데이터를 측정하는 과정에서 트랜잭션의 해시값 뿐만 아니라 수수료 및 크기를 측정할 것이고 메모리풀에서 트랜잭션이 추가되고 제거되는 현상을 더 확실하게 관찰하기 위해 메모리풀의 전체 사이즈, 전체 트랜잭션의 사이즈, 가상 트랜잭션 사이즈 등의 메모리풀에 관한 데이터를 측정할 것이다. 또한 프로그램의 로그를 분석하여 고아 블록(Orphan Block) 또는 폐기된 블록(Stale Block)의 생성 시점 등을 모니터링이 필요하다.[15] 또한 더욱 더 정밀한 값들의 변화를 관찰하기 위해 데이터 수집 주기를 10 분에서 1 분으로 단축 시키고, 더욱더 정확한 측정을 위해 측정 기간을 총 100 분에서 200 분 혹은 300 분으로 증가시켜 데이터를 수집할 계획이다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(교육부)의 재원으로 한국연구재단의 지원과(NRF-2018R1D1A1B07050380) 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539, 블록체인의 트랜잭션 모니터링 및 분석 기술개발))

참 고 문 헌

[1] SooHoon Maeng, Hye-yeong Shin*, Daeyong Kim*, Hongtaek Ju. "Analysis of Memory Pool Jacquard Similarity between Bitcoin and Ethereum in the Same Environment." KNOM Review '19-03 Vol.22 No.03, (2019). [Online]. Available: http://www.knom.or.kr/knom-review/v22n3/3.pdf

[2] SAAD, Muhammad, et al. Exploring the attack surface of blockchain: A systematic overview. arXiv preprint arXiv:1904.03487, 2019.

[3] KOOPS, David. Predicting the confirmation time of bitcoin transactions. arXiv preprint arXiv:1809.10596, 2018.

[4] AL-SHEHABI, Abdullah. Bitcoin Transaction Fee Estimation Using Mempool State and Linear Perceptron Machine Learning Algorithm. 2018.

[5] NAGAYAMA, Ryunosuke; SHUDO, Kazuyuki; BANNO, Ryohei. Simulation of the Bitcoin Network Considering Compact Block Relay and Internet Improvements. arXiv preprint arXiv:1912.05208, 2019.

[6] M. Corallo, Compact Block Relay (BIP 152), [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0152.mediawiki, Accessed: April. 02. 2020

[7] Kyungchan Ko, ChaeHyeon Lee, James Won-Ki Hong. "An Analysis on Similarity of mempool on Bitcoin nodes", KNOM Conference 2019, [Online]. Available: http://dpnm.postech.ac.kr/papers/KNOM/19/2019KNOMConfProc_v1.pdf.

[8] SAAD, Muhammad, et al. Mempool Optimization for Defending Against DDoS Attacks in PoW-based Blockchain Systems. In: 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2019. p. 285-292.

[9] IMTIAZ, Muhammad Anas, et al. Churn in the Bitcoin Network: Characterization and impact. In: 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2019. p. 431-439.

[10] BitcoinCore, Download, [Online]. Available: https://bitcoin.org/en/bitcoin-core/, Accessed: April. 02. 2020.

[11] PARK, Sehyun, et al. Nodes in the bitcoin network: comparative measurement study and survey. IEEE Access, 2019, 7: 57009-57022.

[12] Stanford InfoLab, "Finding Similar Items", [Online]. Available: http://infolab.stanford.edu/~ullman/mmds/ch3a.pdf

[13] DECKER, Christian; WATTENHOFER, Roger. Information propagation in the bitcoin network. In: IEEE P2P 2013 Proceedings. IEEE, 2013. p. 1-10.

[14] Blockchain.com, "Average Transactions Per Block", [Online]. Available: https://www.blockchain.com/en/charts/n-transactions-per-block, Accessed: April. 02. 2020

[15] IMTIAZ, Muhammad Anas; STAROBINSKI, David; TRACHTENBERG, Ari. Characterizing Orphan Transactions in the Bitcoin Network. arXiv preprint arXiv:1912.11541, 2019.

다음 블록에 포함되는 비트코인 트랜잭션 예측 연구

고경찬 O*, 이채현 *, 우종수**, 홍원기*

*포항공과대학교 컴퓨터공학과

**포항공과대학교 정보통신대학원

{kkc90, chlee0211, woojs, jwkhong}@postech.ac.kr

A Study on Predicting Bitcoin Transaction Included in the Next Block

Kyungchan Ko O*, ChaeHyeon Lee*, Jongsu Woo**, James Won-Ki Hong*

*Department of Computer Science and Engineering, POSTECH

**Graduate School of Information Technology, POSTECH

요 약

데이터의 양이 많아지고 분석 기술이 발달하면서 과거의 축적된 데이터를 분석하여 미래에 일어날 일들을 예측하려는 시도들이 생겨나고 있다. 비트코인은 블록체인 기술을 기반으로 최초로 구현된 암호화폐이다. 이 암호화폐는 투명성이 적용된 공개 분산 원장이기 때문에, 누구나 비트코인 데이터에 접근하여 활용할 수 있다. 따라서 비트코인에서 생성된 과거 데이터를 분석하여 비트코인 플랫폼에서 향후에 발생할 이벤트 예측을 시도할 수 있다. 비트코인에서 트랜잭션이 블록에 포함된다는 것은 블록체인 원장에 저장된다는 중요한 의미를 가지고 있지만, 일반 사용자들은 어떤 트랜잭션이 다음 블록에 포함될 것인지 알 수 없다. 본 연구에서는 블록에 포함되는 비트코인 트랜잭션들의 특징을 분석하여 다음에 생성될 블록에 포함될 트랜잭션을 예측하는 실험의 결과를 보여준다. 더 나아가 포함될 만한 트랜잭션이 블록에 포함되지 않는 경우를 조사한다.

I. 서 론

비트코인 [1]은 P2P(peer-to-peer) 네트워크 구조로 구축되어 탈중앙화방식으로 운영되는 전자결제 시스템이다. 비트코인 시스템의 사용자들은 비트코인(화폐)을 다른 사용자에게 전달하기 위해서 이를 위한 트랜잭션(Transaction)을 만들어서, 자신과 연결된 피어(Peer)들에게 브로드캐스트(Broadcast) 한다. 이 트랜잭션을 수신한 피어는 트랜잭션 검증을 수행한 후 그와 연결된 다른 피어들에게 트랜잭션을 전파한다. 이렇게 P2P 네트워크에서 메시지를 전파하는 방식으로 트랜잭션은 비트코인 네트워크 전역으로 전달되게 된다. 하지만 P2P 방식으로 데이터가 전파되면 전파 지연(Propagation delay) 때문에 각 노드(Node)들이 트랜잭션을 수신하는 시간이 달라지며 심지어 중간에 손실되어 트랜잭션을 수신하지 못할 수도 있다. [2]

비트코인 네트워크의 각 노드들이 보유하는 원장은 모두 동일 해야하며, 이는 원장에 저장될 트랜잭션 순서의 획일화를 의미한다. 비트코인 시스템은 동일한 원장을 유지하기 위해서 작업증명(PoW, Proof of Work) [3]라는 합의 알고리즘을 사용한다. 작업증명은 계산 집약적인 연산을 통해서 가장 빠르게 문제의 정답을 찾은 노드에게 블록을 생성할 권한을 부여하는 것이다. 즉, 노드들이 각자 다른 트랜잭션 셋을 블록에 포함시키고 블록헤더 안에 있는 Nonce 값을 조정하면서 연속적으로 블록의 해시값을 계산하는데, Target value 보다 낮은 해시값을 가장 먼저 찾은 노드의 블록을 새로운 블록으로 인정하는 것이다. 작업증명을 수행하는 것을 마이닝(Mining)이라고 부르며 계산 집약적인 연산을 수행하기 위해서 컴퓨팅 리소스가 필요하고 많은 전력이 소모된다. 때문에 해당 작업의 동기를 부여하기 위해서 블록을 생성할 때 인센티브로서 비트코인(블록보상, 트랜잭션 수수료)이 제공된다. 현재는 보상을 얻기 위한 경쟁이 치열해져서 난이도가 많이 증가된 상태이기 때문에, 혼자서는 블록 마이닝에 성공하는 것이 불가능하다. 그래서 여러 마이너(Miner)들이 참여하는 마이닝 풀(Mining pool) [4]을 구성하여 함께 마이닝을 수행한다.

데이터 양이 점점 방대해지고 분석 기술이 발달함에 따라서 과거 데이터를 분석하여 향후에 발생할 이벤트를 예측하려는 시도가 많아지고 있다 [5, 6]. 비트코인 트랜잭션 데이터도 누구에게나 공개되어 있고, 과거의 트랜잭션 데이터를 분석함으로써 비트코인 네트워크에서 향후에 일어날 일을 예측할 수 있다. 현재 새 블록의 생성은 거의 마이닝 풀들

에 의해서 이루어지기 때문에 블록에 포함될 트랜잭션들은 마이닝 풀 오퍼레이터가 결정하고 있다고 볼 수 있다. 그래서 일반 노드들은 다음에 어떠한 트랜잭션이 블록에 포함되는지 정확하게 알 수 없다.

다음 블록에 포함될 트랜잭션을 알 수 있으면 다음과 같은 이점들이 있다. 첫째, 고액의 비트코인이 거래소 주소로 이동하는 것은 많은 양의 비트코인이 명목화폐(USD, KRW)로 교환 될 것이라고 생각할 수 있다. 이것은 비트코인 가격에 영향을 미칠 것이고, 투자자들은 이러한 트랜잭션이 다음 블록에 포함될 것인지 파악함으로 트레이딩을 좀 더 수월하게 할 수 있다. 둘째, 블록에 포함될 것이라고 생각한 트랜잭션이지만 마이닝 풀에서 포함을 시키지 않은 트랜잭션들을 조사할 수 있다. 이것은 마이닝 풀이 올바르게 트랜잭션들을 블록에 포함시키고 있는지 검사하는데 사용할 수 있다. 본 연구는 과거의 멤풀 데이터를 분석해서 다음 블록에 포함되는 트랜잭션들을 예측할 수 있다는 것을 실험을 통해서 보여준다.

II. 관련 연구

Abdullah Al-Shehabi [7]는 원하는 시간에 트랜잭션이 컨펌(Confirm)되기 위해서 필요한 feerate (fee/size)을 예측하기 위해서 weight vector 와 함께 멤풀 상태정보를 사용한 기존과는 다른 접근법을 도입했다. 이때 저자는 트랜잭션이 포함되어야 할 여러 개의 블록 범위들에 대한 weight vector 들을 생성하기 위해서 perceptron machine learning 알고리즘을 사용했다.

Beltran Fiz [8]는 마이닝 풀의 트랜잭션 선택 정책에 변화를 탐지하는 방법을 제안했다. 저자들은 트랜잭션 선택 정책을 일종의 분류 문제로 간주했다. 이 연구에서는 그들의 시나리오를 기반으로 마이닝 풀에서 정책이 변한 것이 어떻게 탐지될 수 있는지 보여줬다.

본 논문의 저자 [9]는 비트코인 멤풀에 있는 트랜잭션들 중에서 다음 블록에 어떤 트랜잭션들이 포함될지 예측하는 연구를 수행했다. 본 논문은 이를 확장하는 연구로서 다양한 분류 알고리즘을 사용해서 더 높은 정확도를 보였고, 추가적으로 블록에 포함될 것이라고 예측했지만 포함되지 않았던 트랜잭션들을 분석했다.

III. 비트코인 모니터링 및 분석 시스템

본 연구는 데이터 수집, 데이터 전처리, 데이터 분석 세 단계로 진행되며, 이를 수행하기 위해서 비트코인 모니터링 및 분석 시스템을 구현했다. 아래의 그림 1 은 비트코인 모니터링 및 분석 시스템의 아키텍처를 보여준다.

비트코인 원장에 저장되는 데이터는 히스토리컬(Historical) 트랜잭션들 이다. 즉, 블록에 포함된 트랜잭션들만 저장되게 된다. 하지만 해당 연구를 수행하기 위해서는 과거에 멤풀이 어떤 상태였는데 어떤 트랜잭션이 블록에 포함되었다는 정보가 필요하기 때문에, 실시간으로 멤풀 상태 데이터를 수집하는 것이 중요하다.

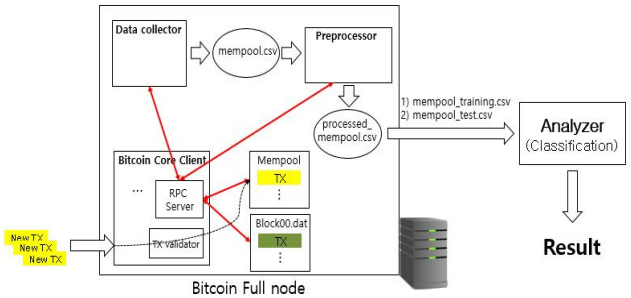


그림 1. 비트코인 모니터링 및 분석 시스템

- **Data collector:** Data collector 는 RPC(Remote Procedure Call) [10] 기반으로 비트코인 네트워크를 모니터링하여 블록(Block), 트랜잭션, 멤풀 상태 데이터를 수집한다. 풀 노드에서 동작하는 Bitcoin core 로 RPC 을 보내서 수신한 응답을 정리하여 csv 파일로 저장한다. 이때 저장하는 멤풀 상태 데이터를 다음에 생성되는 블록의 크기를 임계치(Threshold)로 설정해야 하는데, 비트코인 플랫폼에서 가끔 트랜잭션이 없는 Empty block [11]과 블록의 최대 크기에 훨씬 못 미치는 Abnormal block 이 있기 때문이다. 이러한 일반적이지 않은 데이터를 배제하여 데이터를 수집한다.

- **Data preprocessor:** Data preprocessor 는 수집한 멤풀 상태 데이터에 담겨있는 실시간 트랜잭션들과 생성된 다음 블록에 포함된 트랜잭션들을 비교하여 어떠한 트랜잭션이 블록에 포함 되었는지에 대한 레이블(Label)과 멤풀과 관련된 추가정보를 구한다. Data collector 에서 데이터를 수집할 때는 해당 트랜잭션이 다음 블록에 포함되어 있는 여부를 알 수 없기 때문에 전처리과정에서 이를 분류해주어야 한다. 또한, 비트코인은 UTXO [12]기반의 시스템으로서 트랜잭션의 입력 정보는 원장을 검색해서 얻어야 하며, 이 때문에 Request 들이 추가적으로 발생하게 된다. 그래서 블록 안에 수많은 트랜잭션이 포함되어 있으면 데이터를 수집하는 도중에 실시간으로 해당 트랜잭션들의 디테일 정보를 구하기 어렵다. 아래 표 1 은 전처리 된 데이터에 포함되는 요소들을 설명한다.

Data	Description
txid	transaction id
fee	transaction fee in BTC
size	virtual transaction size as defined

	in BIP 141
feerate	fee / size
txchainCnt	number of related (connected) transactions in mempool
txchainFee	sum of fees of related (connected) transactions in mempool
txchainSize	sum of sizes of related (connected) transactions in mempool
txchainFeerate	txchainFee / txchainSize
timepast	past time after a transaction enters in mempool
num-inputs	number of inputs in a transaction
num-outputs	number of outputs in a transaction
sum-inputs	sum of input values in a transaction
sum-outputs	sum of output values in a transaction
tps	transaction per second
memsize	mempool size
prebs	size of previous block
avg-feerate	feerate on average in mempool
pos-feerate	position of feerate in mempool (in descending order)
c-avg-feerate	sum of feerates of related transactions on an average in the mempool
c-pos-feerate	position of the sum of feerates of related transactions in mempool (in descending order)
next-height	height of the next block
label	included in the next block(1) or not(0)

표 1. 전처리 된 데이터 셋

- **Analyzer:** Analyzer 는 Data preprocessor 를 통해서 전처리 된 Training dataset 과 Testing dataset 을 입력으로 받는다. 실시간 트랜잭션이 다음 블록에 포함될 것인가 여부는 포함(1)되거나 미포함(0)되는 Binary classification 문제로 해석될 수 있기 때문에, Analyzer 는 분류 문제를 해결할 수 있는 머신러닝 모델들(Distributed Random Forest [13], Extremely Randomized Trees [14], Gradient Boosting Machine [15])을 사용하여 결과값을 도출한다. 입력 받은 Training dataset 을 이용해서 위에서 언급한 각 머신러닝 모델들을 학습하고, Test dataset 으로 학습된 모델의 성능을 평가한다.

IV. 실험

비트코인 모니터링 및 분석 시스템으로 데이터를 수집하고 분석을 진행했다. 실험 환경은 Dell PowerEdge R610 서버(Intel(R) Xeon(R) CPU X5650 @ 2.67GHz, 48G RAM)이다. 서버에는 Bitcoin Core 클라이언트(Bitcoin Core 0.17.0 version) [16]를 구동시켜 하나의 노드로서 비트코인 네트워크에 참여했다. Training dataset 을 만들기 위해서 블록 번호 607,702~608,304 에 걸쳐서 데이터를 수집했고, 포함된 트랜잭션이 총 1,927,843 개 이다. Test dataset 은 블록 번호 608,327 ~ 608,345 에 걸쳐서 수집된 데이터로 구성되어있고, 포함된 트랜잭션이 총 140,584 개 이다.

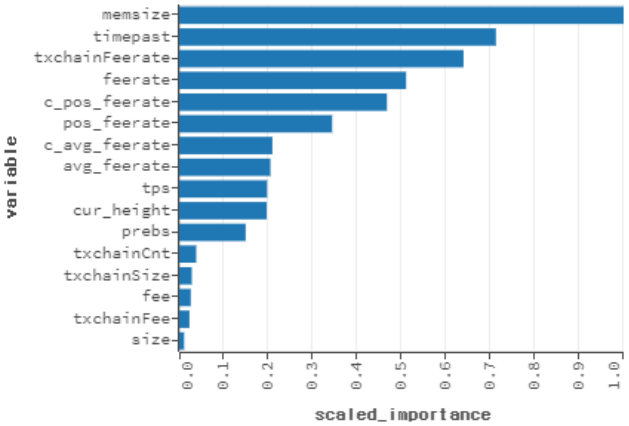
1. 트랜잭션 포함 예측 결과

위에서 설명한 데이터를 기반으로 Analyzer 에서 Distributed Random Forest (DRF), Extremely Randomized Trees (XRT), Gradient Boosting Machine (GBM) 모델을 학습시키고 예측의 성능을 평가했다. 아래의 표 2 는 실시간으로 생성되어 멤풀에 보관되고 있던 트랜잭션들 중에서 다음 블록에 포함될 트랜잭션들을 예측한 실험의 결과를 보여준다.

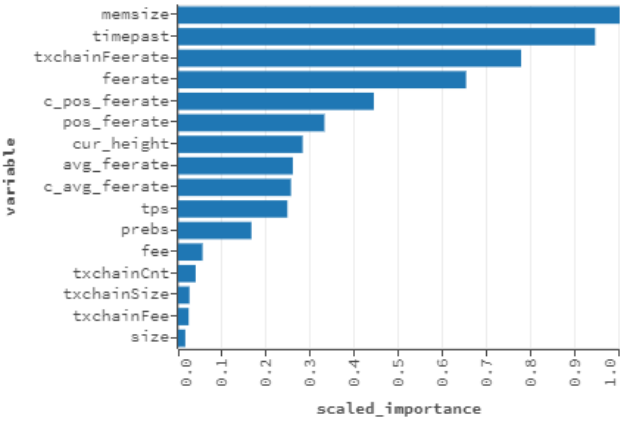
	Precision	Recall	Accuracy	F1-score
DRX	0.8204	0.8960	0.9187	0.8581
XRT	0.8350	0.8688	0.9179	0.8547
GBM	0.8325	0.8697	0.9173	0.8495

표 2. 각 모델에 대한 예측 결과

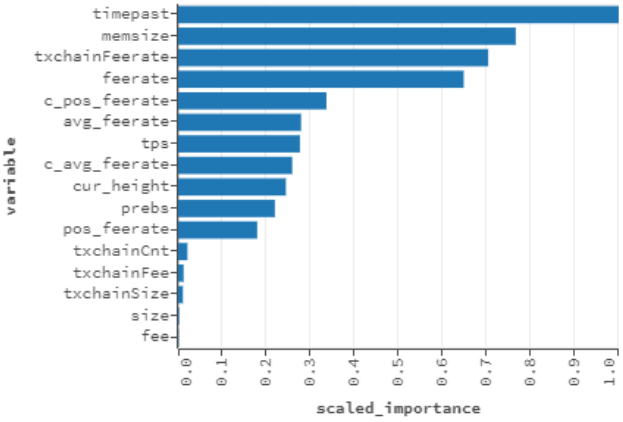
표 2 의 결과를 보면 전처리 된 데이터를 이용해서 학습한 DRX, XRT, GBM 모델 모두 높은 정확도를 보인다는 것을 알 수 있다. 이것은 수집한 데이터를 이용해서 다음 블록에 포함될 트랜잭션을 어느정도 예측할 수 있다는 것을 의미한다. 다음 그림 2 은 각 분류 모델의 Feature Importance [17]를 보여준다. 각 그래프를 보면 해당 모델에서 분류 결과를 도출하는데 있어서 학습한 데이터에서 어떠한 특징이 주요한 영향을 미쳤는지 확인할 수 있다.



(가) DRF 의 Feature Importance



(나) XRT 의 Feature Importance



(다) GBM 의 Feature Importance

그림 3. 각 분류 모델의 Feature Importance 결과

각 모델의 Feature Importance 결과를 보면 상위 5 개의 특징이 동일하다. 즉, 다음 블록에 트랜잭션이 포함되는지 여부는 memsize, timepast, txchainFeerate, feerate, c_pos_feerate 특징들에 영향을 가장 많이 받는다는 것을 확인했다. 블록의 최대 크기가 제한되어 있기 때문에 블록이 생성될 당시 멤풀에 저장되어 있는 트랜잭션의 수가 많을 수록 더 다음 블록에 포함되기 어려운데, memsize 는 이러한 특징이 많이 영향을 끼치는 것을 나타내고 있다. Fee 는 마이너가 수익을 얻는 수단중에 하나이기 때문에 feerate 이 높은 트랜잭션을 우선으로 블록에 포함시킨다는 것을 나타낸다. 이것은 txchainFeerate, feerate, c_pos_feerate 의 높은 Feature Importance 가 설명해주고 있다. 멤풀에 먼저 들어온 트랜잭션일수록 마이너가 블록에 포함될 트랜잭션을 선정하는데 우위를 차지하는데, timepast 의 높은 feature Importance 가 이를 나타내고 있다.

2. 트랜잭션 배제 분석

트랜잭션이 다음 블록에 포함될지 예측한 실험에서 timepast 와 txchainFeerate 는 높은 Feature Importance 를 보였다. 이 중요한 특징들을 이용하여 역으로 다음 블록에 포함될 것으로 예측되었지만 포함이 안된 트랜잭션에 대해서 분석을 하였다. Filtering 의 첫째 조건은 블록에 포함될 것이라고 예측된 트랜잭션들의 txchainFeerate 의 중간값보다 높은 값을 갖는 트랜잭션이 다음 블록에 포함되지 않은 경우 이다. 둘째 조건은 멤풀에 들어온 이후로 경과된 시간이다. 본 실험에서 기준 시간 제약은 임의로 20 분으로 설정했다. Filtering 의 목적은 충분히 높은 txchainFeerate 을 가지고 있고 멤풀에 들어 온지 충분히 긴 시간인 20 분이 지났지만 아직 다음 블록을 통해서 생성되지 않은 트랜잭션을 추출하는 것이다. 아래 그림 4 는 해당 분석 내용을 그래프로 그린 것이다. 그림 4 의 빨간색 점은 앞에서 언급한 filtering 조건 두개를 모두 만족하는 트랜잭션 들이다. (1), (2) 부분을 보면 다음 블록에 포함될 가능성이 높은 많은 수의 트랜잭션들이 실제로는 마이닝 풀에 의해서 블록에 포함되지 않았다는 것을 생각해 볼 수 있다. (1)은 각각 607798, 607798 번째블록이다. (2)는 각각 607810, 6078111, 707812 번째 블록을 의미한다. 공교롭게도 (1), (2)의 블록들은 모두 Poolin 이라는 마이닝 풀에서 채굴하였다.

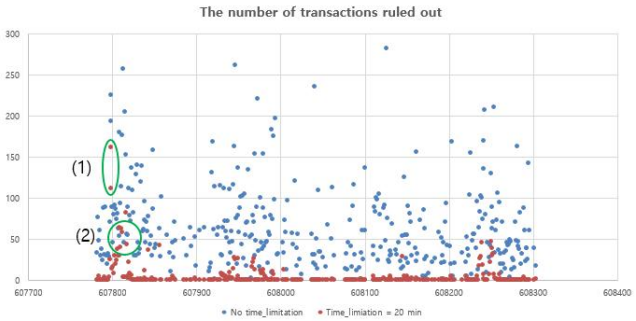


그림 4. 예측이 잘못된 트랜잭션 분석

분석된 내용만 봐서는 특정 마이닝 풀의 오퍼레이터가 의도적으로 filtering 된 트랜잭션들을 배제했는지 알 수 없다. 하지만 해당 분석을 통해서 정상적으로 블록에 포함되어야 하는 트랜잭션들이 얼만큼 배제되고 있는지 확인할 수 있다.

V. 결론 및 향후 연구

비트코인에서 마이너가 최대의 수익을 얻기 위해서 크기당 수수료(fee/size)가 높은 트랜잭션을 최대한 많이 블록에 포함시켜야 하기 때문에, 일반적으로 feerate 이 높은 트랜잭션들이 우선으로 블록에 포함된다고 알고있다. 본 연구에서는 비트코인 네트워크에서 수집한 과거 멤풀 상태 정보와 히스토리컬 트랜잭션 정보를 분석하여, 어떤 특징들이 트랜잭션이 다음 블록에 포함 될지를 결정하는데 많은 영향을 미치는지 확인했다. 분석 결과 memsize,

timepast, txchainFeerate, feerate, c_pos_feerate 특징들이 가장 많은 영향을 미치는 요소임을 알 수 있었다. 또한, 수집한 데이터를 학습한 세 종류의 분류 모델들 모두 90%이상의 높은 예측 정확도를 보여주었다. 향후 연구로는 여러 노드의 멤풀 정보를 수집하여 실험을 확장할 예정이며 Deep learning 알고리즘을 사용한 결과와 예측 정확도를 비교해볼 예정이다.

ACKNOWLEDGMENT

본 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 임 (No.2018-0-00539)

참 고 문 헌

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).
[2] 고경찬, 이채현, 홍원기, "비트코인 노드 메모리 풀 유사도 분석", KNOM Conference 2019, Daegu, Korea, May. 30, 2019, pp. 16-18.
[3] Zheng, Zibin, et al. "An overview of blockchain technology: Architecture, consensus, and future trends." 2017 IEEE international congress on big data (BigData congress). IEEE, 2017.
[4] “Mining pool”, https://en.wikipedia.org/wiki/Mining_pool, Accessed: April 1, 2020.
[5] Bunker, Rory P., and Fadi Thabtah. "A machine learning framework for sport result prediction." Applied computing and informatics 15.1 (2019): 27-33.
[6] Yan, Xiang-Bin, et al. "Research on event prediction in time-series data." Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826). Vol. 5. IEEE, 2004.
[7] Wang, Aiping, et al. "An incremental extremely random forest classifier for online learning and tracking." 2009 16th ieee international conference on image processing (icip). IEEE, 2009.
[7] Al-Shehabi, Abdullah. "Bitcoin Transaction Fee Estimation Using Mempool State and Linear Perceptron Machine Learning Algorithm." (2018).
[8] Pontiveros, Beltran Borja Fiz, Robert Norvill, and Radu State. "Monitoring the transaction selection policy of Bitcoin mining pools." NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2018.
[9] Ko, Kyungchan, et al. "Prediction of Bitcoin Transactions Included in the Next Block." International Conference on Blockchain and Trustworthy Systems. Springer, Singapore, 2019.
[10] Nelson, Bruce Jay. "Remote procedure call." (1981).
[11] “Empty block”, https://www.blockchain.com/ko/btc/block/625377, Accessed: April 11, 2020.
[12] “UTXO”, https://bitcoin.org/en/glossary/unspent-transaction-output, Accessed: April 11, 2020.
[13] “Distributed Random Forest”, http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/drf.html, Accessed: April 11, 2020.
[14] Wang, Aiping, et al. "An incremental extremely random forest classifier for online learning and tracking." 2009 16th ieee international conference on image processing (icip). IEEE, 2009.
[15] Friedman, Jerome H. "Greedy function approximation: a gradient boosting machine." Annals of statistics (2001): 1189-1232.
[16] “Bitcoin Core 0.17.0. released”, https://bitcoin.org/en/release/v0.17.0, Accessed: April 11, 2020.
[17] “Variable importance”, http://docs.h2o.ai/h2o/latest-stable/h2o-docs/variable-importance.html, Accessed: April 11, 2020.

블록체인 기반 전자 투표 시스템의 프론트엔드 설계 및 구현

김보선, 김태연, 신무곤, 백의준, 김명섭
고려대학교

{boseon12, ingcubators, tm0309, pb1069, tmskim}@korea.ac.kr

Front-End Design and Implementation of Blockchain based Electronic Voting System

Boseon Kim, Tae-Yuen Kim, Mu-Gon Shin, Uijun Baek, Myung-Sup Kim
Korea Univ.

요 약

오프라인 투표는 큰 비용과 인력을 필요로 하며 기존의 온라인 투표는 중앙 집중적 구조로 신뢰성이 떨어지는 여러 문제를 야기한다. 이러한 문제들은 블록체인 기반 전자 투표 시스템을 통해 해결할 수 있다. 본 논문은 오프라인 투표와 기존의 온라인 투표의 문제를 개선한 블록체인 기반 전자 투표 시스템의 웹 프론트엔드를 구현하고 세부 통신 과정에 대해 설명한다.

Ⅰ. 서론

최근 몇몇 대학의 총학생회 선거에서 예정된 기간 동안 개표 가능한 투표율을 넘지 못해 재투표를 하는 상황이 빈번하게 발생하고 있다. 오프라인 투표는 투표소에 직접 가야 하는 불편함이 존재하고 많은 비용과 시간이 소모되며[1] 잉크 번짐이나 두 표를 행사하는 행위로 인해 무효 표가 발생할 수 있다. 이와 같은 문제점들은 전자 투표 시스템으로 해결할 수 있다. 전자 투표는 유권자가 스마트폰이나 컴퓨터와 같은 온라인 시스템을 통해 투표하는 방식으로 시간과 공간의 제약이 없어 유권자들에게 편의를 제공하고 참여율과 투표율 상승을 기대할 수 있다. 하지만 중앙 집중적 구조로 인해 청탁 등 부정행위가 발생할 수 있다. 따라서 블록체인을 활용한 관리 서버나 주체가 없는 탈 중앙화 시스템이 필요하다[2].

본 논문은 이더리움을 이용한 전자 투표 시스템의 웹 프론트엔드 구조를 제안한다. 제안한 프론트엔드 시스템은 컴포넌트 기반의 확장성, 결합성, 재사용성을 제공하는 ReactJS 를 사용한다.

본 논문은 서론에 이어 본론에서 블록체인 기반 전자 투표 시스템의 각 컴포넌트별 통신 과정을 설명하고 결론에서는 해당 연구를 정리하고 향후 연구 방향을 제시한다.

Ⅱ. 관련 연구

[1]은 유무선 연동의 모바일 전자 투표 시스템을 설계하고 구현하였다. 이는 투표의 번거로움과 수업 결손을 최소화하고, 학교 경영의 의사 결정에서 학부모와 학생 참여를 위한 도구가 되었다.

[2]는 블록체인을 기반으로 하는 이더리움 환경에서 전자 투표 시스템을 구축했으며 전자 투표의 기본적인 요구 사항을 제시하고 해당 요구 사항에 따라 시스템을 평가하였다.

Ⅲ. 본론

본 장은 블록체인 기반 전자 투표 시스템의 라우터 구조와 프론트엔드의 통신 과정을 설명한다. 홈 컴포넌트는 회원가입, 로그인, 투표 생성, 투표, 투표 결과 확인 컴포넌트를 라우팅하며 구조는 그림 1 과 같다.

그림 1. 라우터 구조

다음으로 우리는 각 컴포넌트별 통신 과정에 대해 설명한다.

1) 회원가입 과정

사용자는 회원가입 페이지의 입력 폼을 통해 아이디, 비밀번호, 이름을 입력하고 회원가입 컴포넌트는 입력받은 3 개의 개인 정보를 백엔드로 전송한다. 백엔드는 데이터베이스 내 저장된 개인 정보를 조회하고 기존 데이터가 존재하지 않으면 데이터베이스에 개인 정보를 저장하고, 존재하면 에러 값을 반환한다. 회원가입 컴포넌트는 개인 정보 조회 결과에 따라 사용자에게 성공 혹은 실패 메시지를 전달한 후 로그인 페이지로 라우팅한다. 회원가입 컴포넌트의 통신 과정은 그림 2 와 같다.

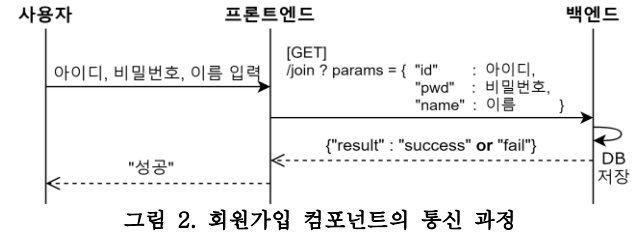


그림 2. 회원가입 컴포넌트의 통신 과정

이 논문은 2018 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539-003,블록체인의 트랜잭션 모니터링 및 분석 기술개발)

2) 로그인 과정

사용자는 로그인 페이지의 입력 폼을 통해 아이디와 비밀번호를 입력하고 로그인 컴포넌트는 입력받은 개인 정보를 백엔드로 전송한다. 백엔드는 수신한 개인 정보와 데이터베이스 내 저장된 개인 정보를 비교하고 아이디와 비밀번호가 일치하면 로그인 컴포넌트에 성공 혹은 실패 여부를 전달한다. 로그인 컴포넌트는 전달받은 값에 따라 사용자에게 성공 혹은 실패 메시지를 보여준다. 로그인 컴포넌트의 통신 과정은 그림 3 과 같다.

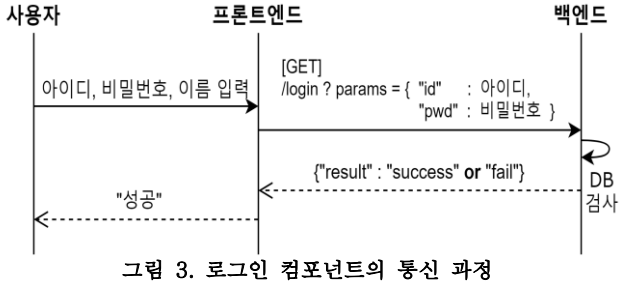


그림 3. 로그인 컴포넌트의 통신 과정

3) 투표 생성 과정

투표 생성 페이지는 투표 목록을 생성한다. 관리자는 입력 폼을 통해 투표 제목과 투표 항목들을 입력하고 백엔드로 전송한다. 투표 생성 컴포넌트는 입력받은 투표 정보를 백엔드로 전송한다. 백엔드는 수신한 투표 제목을 데이터베이스 내에서 검색하고 동일한 투표 제목이 존재하지 않으면 투표 제목과 투표 항목들을 저장하고 투표 생성 컴포넌트에 “success”를 전송한다. 만약 백엔드가 수신한 투표 제목과 동일한 투표 제목이 데이터베이스 내에 존재하지 않으면 백엔드는 투표 생성 컴포넌트에 “fail”을 전송한다. 투표 생성 컴포넌트의 통신 과정은 그림 4 와 같다.

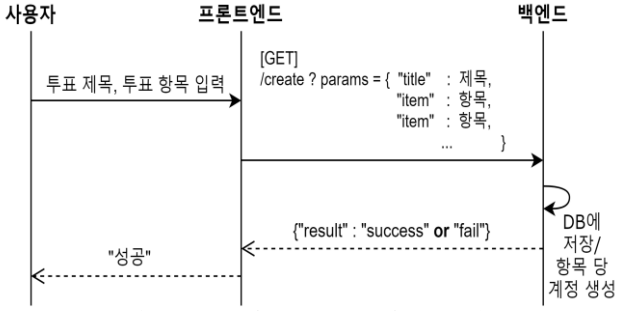


그림 4. 투표 생성 컴포넌트의 통신 과정

4) 투표 과정

투표 컴포넌트는 백엔드로부터 진행 중인 투표의 제목들을 전달받는다. 사용자가 투표 페이지를 통해 자신이 투표하고자 하는 투표 제목을 선택하면, 투표 컴포넌트는 선택된 제목을 백엔드로 전송하고 해당 투표 제목의 투표 항목들을 수신하여 사용자에게 투표 항목들을 보여준다. 사용자가 투표할 항목을 선택하면 투표 컴포넌트는 백엔드로 투표자의 개인 정보, 투표 제목, 선택된 투표 항목을 전달한다. 백엔드는 투표자의 신원을 확인하고 투표자의 계정으로부터 투표 항목, 즉 후보자의 계정으로 1 이더를 전송하는 트랜잭션을 생성한다. 1 이더는 1 표이고 투표 항목 계정의 보유 이더는 해당 투표 항목에 대한 투표수를 의미한다. 투표 컴포넌트의 통신 과정은 그림 5 와 같다.

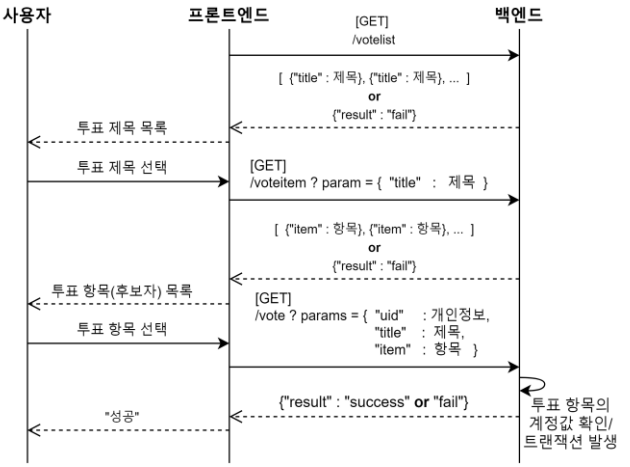


그림 5. 투표 컴포넌트의 통신 과정

5) 투표 결과 확인 과정

투표 결과 확인 컴포넌트는 백엔드로부터 종료된 투표의 제목들을 전달받는다. 사용자가 투표 결과 확인 페이지를 통해 결과를 확인하고자 하는 투표 제목을 선택하면, 투표 컴포넌트는 선택된 제목을 백엔드로 전송하고 해당 투표 제목의 투표 항목들과 항목별 투표수를 수신하여 사용자에게 제공한다. 투표 결과 확인 컴포넌트의 통신 과정은 그림 6 과 같다.

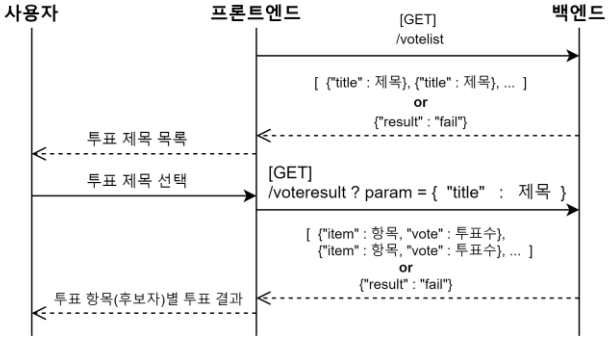


그림 6. 투표 결과 확인 컴포넌트의 통신 과정

Ⅳ. 결론

본 논문은 블록체인 기반 전자 투표 시스템의 웹 프론트엔드 구조를 제안하고 구현하였다. 본 논문에서 제안한 웹 프론트엔드 구조는 손쉽게 활용 및 수정이 가능하며 우리는 블록체인 기반 전자 투표 시스템의 프론트엔드 세부 통신 과정을 자세하게 설명함으로써 구현의 접근성을 높였다. 그러나 제안한 프론트엔드 구조는 [2]에서 정의한 전자 투표의 요구 사항을 모두 만족하지 못하며 단순한 투표에 관한 기능만을 제공하기에 보안 위협에 취약하다. 본 논문에서 사용한 ReactJS 는 구조 및 기능 확장이 용이하여 보안 기능을 포함한 추가 기능의 원활한 구현을 기대할 수 있다. 우리는 향후 전자 투표 시스템의 요구 사항을 모두 만족하는 전자 투표 시스템 구축에 관하여 연구를 수행할 계획이다.

참 고 문 헌

[1] 오필우; 김명렬; 신수범. 학생임원 선출 방법의 개선을 위한 모바일 전자투표 시스템. 정보교육학회논문지, 2006, 10(1), 813-822

[2] 이루다; 임좌상. 블록체인을 활용한 전자투표 시스템 구축. 한국 정보통신학회논문지, 2019, 23(1), 103-100.

블록체인 기반 전자 투표시스템의 백엔드 서비스 구현

김태연, 김보선, 신무곤, 백의준, 김명섭
고려대학교

{ingcubators, boseon12, tm0309, pb1069, tmskim}@korea.ac.kr

Blockchain based E-Voting System Back-End Service Implementation

Tae-Yuen Kim, Boseon Kim, Mu-Gon Shin, Uijun Baek, Myung-Sup Kim
Korea Univ.

요 약

최근 전자 투표 시스템에 관한 관심이 늘어감에 따라 블록체인 기반 전자 투표 시스템에 관한 연구가 늘어가고 있다. 기존의 전자 투표 시스템이 갖는 상호 의존성과 관리자의 과도한 권한 문제, 데이터의 무결성 문제를 가지며, 블록체인 기반 전자 투표 시스템은 이러한 문제점들을 해결할 수 있다. 기존의 블록체인 기반 전자 투표 시스템에 대한 연구는 시스템 구조의 제안과 기존 전자 투표 시스템의 문제 해결에 초점을 맞추고 있다. 그러나 후발 연구자들을 위한 세부 통신 과정이나 구현에 대한 설명은 부족하다. 따라서 본 논문은 블록체인 기반 전자 투표 시스템의 API 를 설계 및 구현하고 세부 통신 과정에 대해 설명한다.

Ⅰ. 서 론

최근 유권자에게 편리성을 제공하고 투표 참여율을 증가시킬 수 있는 전자 투표 시스템에 관하여 많은 연구가 수행되고 있다. 전자 투표 시스템은 시공간의 제약을 받지 않기 때문에 종이투표 방식의 저조한 투표율을 일부 해결했다. 그러나 외부의 공격으로 인해 데이터의 변조와 조작이 쉬워 투표의 무결성이 보장되지 않는 문제점을 가진다. 블록체인 기반 전자 투표 시스템은 투표 결과를 분산하여 저장하므로 기존의 전자 투표 시스템이 갖는 무결성 문제를 해결할 수 있다. 블록체인 기반 전자 투표시스템은 사용자의 입력을 처리하는 프론트엔드와 실제 블록체인 네트워크와 통신하며 투표를 생성하고 관리하는 백엔드로 구성된다. 우리는 사설 블록체인 네트워크를 구축했으며 이에 기반한 블록체인 기반 전자 투표 시스템의 API 를 설계 및 구현하고 백엔드의 세부 통신 과정에 대해서 설명한다.

Ⅱ. 관련 연구

블록체인은 분산 컴퓨팅 기술 기반의 데이터 위변조 방지 기술이며 P2P 방식 기반으로 블록을 체인 형태로 연결하여 데이터베이스에 저장함으로써 데이터 무결성을 제공한다.

[1]은 기존 전자 투표 시스템의 선거원칙 문제를 해결할 수 있는 효율적인 프라이빗 블록체인 기반 전자 투표 시스템에 대해 제안하고 실제로 구현하였다.

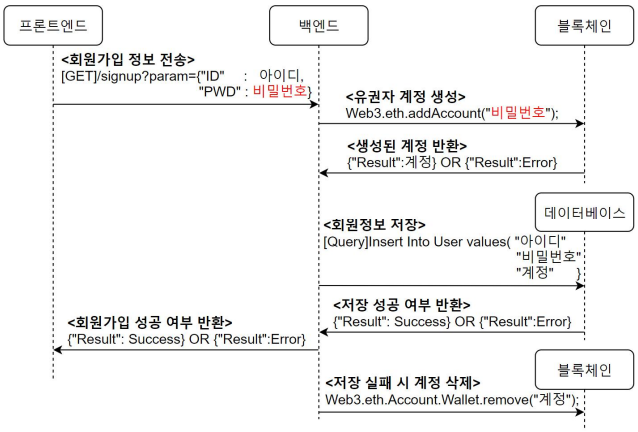
[2]는 캠퍼스 내에서 진행되는 투표를 위한 전자 투표시스템을 제작하고, 보안성을 강화하기 위해 블록체인기술을 활용하였다.

Ⅲ. 본론

본 논문에서 블록체인 기반 전자 투표 시스템 백엔드의 통신 과정은 회원가입, 투표생성, 투표하기 그리고 결과 확인으로 구성된다.

1. 회원 가입 과정

백엔드는 프론트엔드로부터 개인정보를 전달받아 블록체인에 계정생성을 요청하고 성공 혹은 실패 여부를 반환받는다. 계정생성 이후 백엔드는 프론트엔드로부터 전달받은 개인정보와 동일한 데이터가 데이터베이스 내에 존재 하는지 검사하고 존재하지 않으면 해당 개인정보를 저장한다. 데이터베이스에 개인정보를 저장할 때 에러가 발생하면 백엔드는 블록체인에 이미 생성된 계정에 대한 삭제를 요청한다. 회원 가입 과정은 그림 1 에서 표현하고 있다.

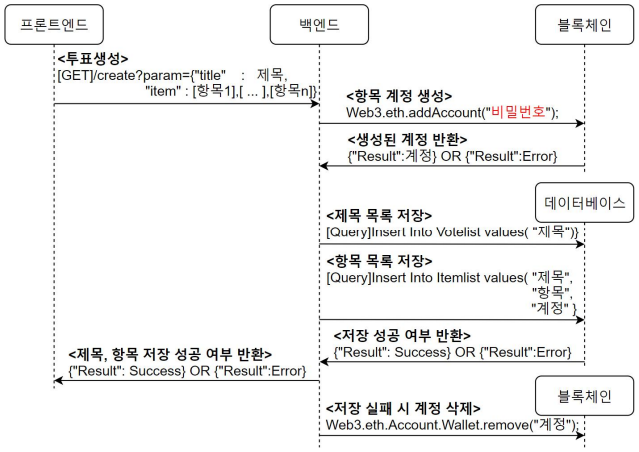


<그림 1> 회원 가입 과정

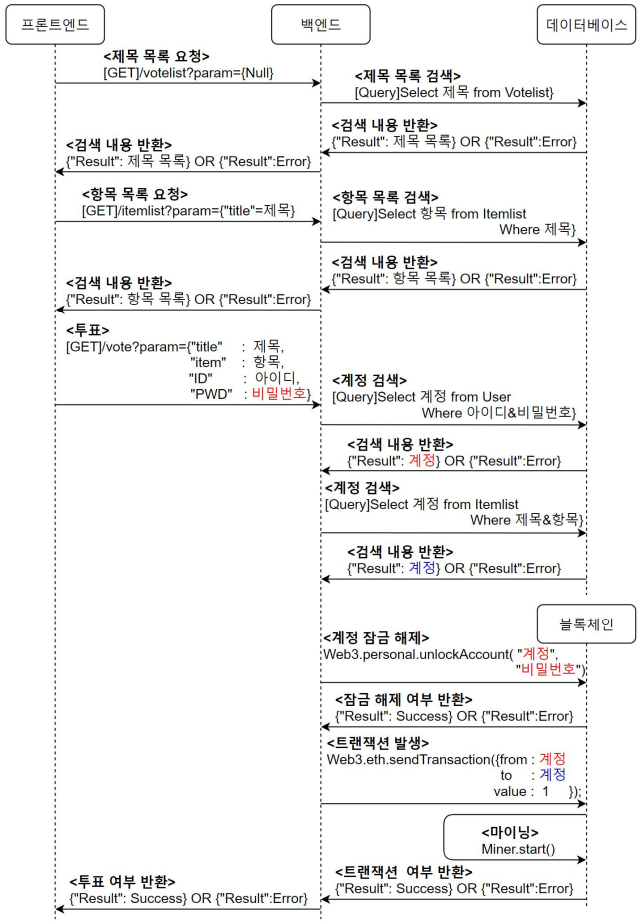
이 논문은 2018 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539-003,블록체인의 트랜잭션 모니터링 및 분석 기술개발)

2. 투표 생성 과정

백엔드는 프론트엔드로부터 투표 제목과 투표 항목들을 전달받아 블록체인에 투표 항목 개수만큼 계정생성을 요청하고 성공 혹은 실패를 반환받는다. 계정생성 이후 백엔드는 프론트엔드로부터 전달받은 투표정보와 동일한 데이터가 데이터베이스 내에 존재하는지 검사하고 존재하지 않으면 해당 투표정보를 저장한다. 데이터베이스에 저장할 때 에러가 발생하면 백엔드는 블록체인에 이미 생성된 계정에 대한 삭제를 요청한다. 투표생성 과정은 그림 2 에서 표현하고 있다.



<그림 2> 투표 생성 과정



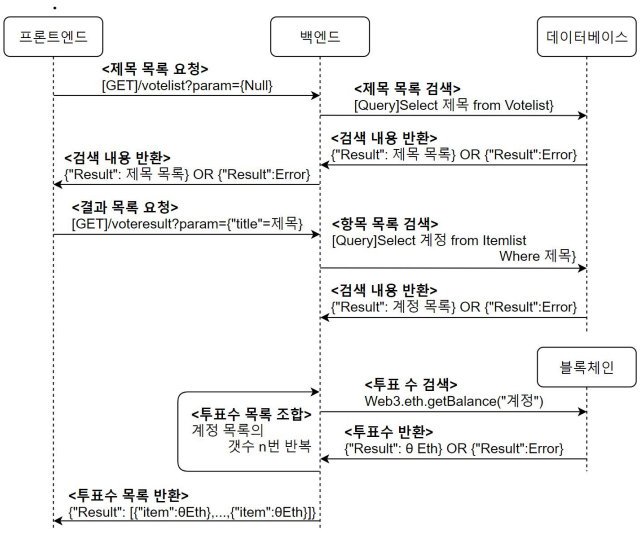
<그림 3> 투표 과정

3. 투표 과정

백엔드는 데이터베이스 내에 저장된 투표목록과 투표제목를 프론트엔드에 전달한다. 다음으로 백엔드는 프론트엔드로부터 개인정보와 투표내역을 전달받아 데이터베이스 내에서 사용자 개인정보에 해당하는 투표자 계정 정보와 투표내역에 해당하는 후보자 계정 정보를 가져와서 투표자 계정으로부터 후보자 계정에 1 이더를 전송하는 트랜잭션을 생성한다. 마지막으로 트랜잭션 생성에 대한 성공 및 실패 여부를 프론트엔드에 전달한다. 투표 과정은 그림 3 에서 표현하고 있다.

4. 결과 확인 과정

백엔드는 데이터베이스 내에 저장된 투표목록을 프론트엔드에 전달한다. 다음으로 백엔드는 프론트엔드로부터 투표내역을 전달받아 투표내역에 해당하는 후보자계정이 현재 보유하고 있는 이더 잔액을 가져온다. 마지막으로 백엔드는 프론트엔드에 투표 항목별 투표수를 전달한다. 결과 확인 과정은 그림 4 에서 표현하고 있다.



<그림 4> 결과 확인 과정

Ⅳ. 결 론

본 논문은 블록체인 기반 전자 투표 시스템의 API 를 설계 및 구현하고 세부 통신 과정에 대해 설명하였다. 우리는 후발 연구자들이 본 논문을 통해 기초적인 블록체인 기반 전자 투표 시스템 백엔드 서비스 구현을 도울 것이라고 기대한다. 그러나 여전히 관리자가 투표내역을 열람할 수 있다는 보안 문제가 존재한다. 따라서 우리는 이러한 보안 문제를 해결 될 수 있는 블록체인 기반 전자투표 시스템에 대해서 연구할 계획이다.

참 고 문 헌

[1] 권용석, et al. 프라이빗 블록체인 기반 전자투표시스템 구현. 2018.

[2] 양준호, et al. "블록체인기술 기반 캠퍼스 e-투표 시스템." 한국컴퓨터교육학회 학술발표대회논문 (2018): 67-70.

악성코드 패킹 탐지에서 기계학습 알고리즘 비교

김종욱°, 남궁주홍, 문양세, 최미정
강원대학교 컴퓨터과학과

{goldbear564, namgung_juhong, ysmoon, mjchoi}@kangwon.ac.kr

Comparison of Machine Learning Algorithms in Malware Packing Detection

Jong-Wouk Kim°, Juhong Namgung, Yang-Sae Moon, Mi-Jung Choi
Department of Computer Science, Kangwon National University

요 약

최신 악성코드는 백신 프로그램을 우회하기 위해 변이, 난독화, 변형 등의 기술을 사용하여 각종 컴퓨터나 장치들을 감염시키고 있다. 또한, 많은 수의 변종된 악성코드가 새롭게 생성 및 탐지되고 있다. 하지만 시그니처 기반 탐지와 같이 전통적인 패킹 탐지 방식은 오래된 데이터베이스를 사용하기 때문에 새롭게 등장한 패커의 경우 변종된 악성코드를 탐지할 수 없다는 단점이 있다. 이러한 문제로 본 논문은 기계학습 기반의 악성코드 탐지 및 분류 연구를 진행하였다. 특히, 본 논문에서는 중요한 특징들을 사용하여 다양한 악성코드의 패킹 여부를 탐지하기 위해 다양한 기계학습 알고리즘을 적용해보고 분류, 탐지 및 비교하였다. 그 결과, 시그니처 기반에서는 탐지할 수 없었던 커스텀 패커(custom packer)로 패킹된 악성코드에 대해서도 잘 동작함을 확인하였다.

Ⅰ. 서 론

악성 소프트웨어는 보안에 있어 지속적으로 증가하고 있는 문제들 중 하나이다. 시스코는 2018 년 평창 겨울 올림픽 개막식이 진행되는 동안 와이파이 및 공식 홈페이지가 다운되는 문제가 발생하였다. 이후 시스코의 조사 결과 악성코드에 의한 문제임이 밝혀졌다[1]. 이러한 대규모 서비스의 중단은 개인 정보 보호 위험, 브랜드 평판 저하, 서비스 품질 저하 등 수많은 문제를 야기한다. 전문가들은 다양한 문제들을 해결하기 위해 신속히 조사한 결과 서버의 자산을 파괴하도록 설계된 악성코드라는 것과 발생지를 숨기고 연구자들을 속일 수 있는 진보된 악성 소프트웨어임을 밝혀냈다. 이처럼 점점 더 많은 수의 악성코드가 온라인에서 생성되고 변종 악성 소프트웨어도 등장하고 있다.

악성 소프트웨어가 급증하면서 효과적인 악성 소프트웨어 탐지 및 분석은 사용자, 서비스 제공자와 시스템 보안의 기본이 되었다. 악성 소프트웨어를 분석하는 방법은 정적 및 동적으로 나눌 수 있다. 정적 분석은 악성 소프트웨어를 실행하지 않고, 악성 소프트웨어의 형태별 특성을 검사 및 추출하여 분석하는 방법이다. 동적 분석은 샌드박스 또는 가상 환경에서 악성 소프트웨어를 직접 실행하여 행위를 관찰한다. 동적 분석은 정적 분석보다 많은 시간과 노력이 필요하기 때문에 정적 분석에서 최대한 많은 정보를 알아내는 것이 중요하다. 하지만 정적 분석을 방해하는 효과적인 기술들 중 악성 소프트웨어를 패킹(packing)하는 방법이 있다. 이 기술은 악성 소프트웨어의 원본 데이터를 압축 및 암호화하여 정적 분석을 사전에 방지 또는 지연시킨다. 이와 같은 샘플의 악성 행위 여부를 탐지 및 분석하기 위해서는 먼저 패킹 여부를 탐지하고, 동적 분석을 수행해야 한다는 시간적 소모가 생긴다. 따라서, 본 논문에서는 효과적으로 패킹 여부를 탐지하기 위해 악성 파일에서 수집한 특징들을 학습시킨 기계학습 알고리즘들을 비교하는 실험을 진행하고, 그 결과를 분석하였다.

본 논문에서는 서론에 이어 2 장에서는 실험에서 사용한 특징들과 학습 데이터의 관해 설명한다. 3 장에서는 제안하는 분류 기법을 적용시켜 다양한 알고리즘의 비교를 위해 분류 정확도와 F1 점수를 측정 및 비교한다. 마지막으로 4 장에서 결론 및 향후 과제에 대해 언급한 뒤 논문을 마친다.

Ⅱ. 본론

본 절에서는 악성코드 패킹 여부 탐지를 위한 기계학습 알고리즘이 학습할 특징들을 설명한다. 본 논문에서는 표 1 에 나타난 특징들을 사용하여 각 기계학습 알고리즘을 학습시켰다. 학습에 사용된 특징은 5 개의 불리언형, 2 개의 실수형, 6 개의 정수형으로 구성된다.

불리언 자료형으로 된 5 개의 특징은 Entry Point Section(EPS) 여부, 패커 시그니처 여부, EPS 의 '쓰기(WRITE)' 속성 여부, [2]에서 언급한 범위에 EPS 엔트로피 수치의 포함 여부, [3]에서 언급한 범위에 속한

파일 엔트로피 수치의 포함 여부가 있다. EPS 여부는 Entry Point 가 나타내는 주소가 EPS 의 범위 내에 있다면 0, 없다면 1 을 기록한다. 두 번째로 패커 시그니처가 있다면 1, 없다면 0 을 기록했으며, 세 번째로 EPS 에 쓰기 속성이 있는지 참조하여 있다면 1, 없다면 0 을 기록한다. 네 번째로 [2]에서 언급한 범위에 EPS 의 엔트로피 수치가 포함되면 1, 포함되지 않는다면 0 을 나타내며, 다섯 번째로 [3]에서 언급한 범위에 엔트로피 수치가 포함되면 1, 그렇지 않다면 0 을 기록한다.

실수형으로는 계산된 EPS 와 파일의 엔트로피 수치를 특징으로 같이 기록하였다. 뿐만 아니라, Portable Executable(PE) 기반 악성코드에 존재하는 섹션들의 정보를 추출했다. 컴파일에 의해 생성되는 일반적인 섹션(e.g. `.text`, `.data`, etc.)의 수, 일반적이지 않은 섹션(e.g. `.upx0`, `.nsp0`, etc.)의 수, 전체 섹션들 중 쓰기 속성을 갖는 섹션들의 수, 전체 섹션들 중 읽기, 쓰기, 실행 속성을 갖는 섹션들의 수, 전체 섹션들 중 실행 속성을 갖는 섹션들의 수와 마지막으로 Import Address Table(IAT)에 있는 함수들의 수를 추출하여 총 13 개의 특징들을 학습하도록 설계했다. 표 1 의 특징들을 학습한 모델의 성능을 분류 정확도와 F1 점수로 표현함으로써 각 모델의 효율성을 비교하였다.

표 1. 학습에 사용된 특징.

Feature	Range of Values
EPS(Entry Point Section)	0 ∨ 1
Packer	0 ∨ 1
WRITE Attribute of EPS	0 ∨ 1
EPS Entropy TF	0 ∨ 1
File Entropy TF	0 ∨ 1
EPS Entropy	[0.00, 8.00]
File Entropy	[0.00, 8.00]
Number of Standard Section	Integer ≥ 0
Number of non-Standard Section	Integer ≥ 0
Number of WR Section	Integer ≥ 0
Number of R/W/E Section	Integer ≥ 0
Number of Executable Section	Integer ≥ 0
Number of Functions in IAT	Integer ≥ 0

Ⅲ. 실험 결과

본 절에서는 표 1 의 특징들로 학습하여 다양한 기계학습 알고리즘에 적용한 결과를 정리했다. 사용된 기계학습 알고리즘은 Decision Tree, Random Forest, SVM, KNN, 다층 퍼셉트론을 사용하였다. 이때 사용된 다층 퍼셉트론의 구성은 그림 1 과 같다. 이 인공 신경망은 13 개의 특징을 64 개의 뉴럴을 가지는 입력 레이어부터 12 개의 히든 레이어와 1 개의 뉴럴을 가지는 출력 레이어로 구성했다.

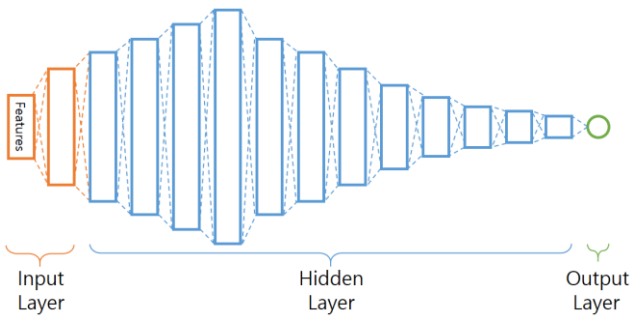


그림 1. 악성코드 패킹 탐지를 위한 인공 신경망.

다양한 기계학습 알고리즘을 적용한 악성코드 패킹 탐지 기법의 적합성을 검증하기 위해 각 알고리즘 별 실험을 진행했다. 실험에 사용된 데이터 셋으로는 총 10,006 개의 파일들을 사용했으며, 직접 패킹한 파일, VirusShare 로부터 수집한 악성코드, 운영체제(Windows 7)에서 수집한 PE 파일들을 사용했다. 또한 악성코드의 패킹 여부를 탐지하기 위해 [2]에서 사용한 방법으로 악성코드의 패킹 여부를 레이블링하여 본 모델에 학습을 시킨다. 레이블링된 데이터로부터 13 개의 특징을 추출하고, 추출된 데이터를 5 가지의 기계학습 알고리즘으로 정확도와 F1 점수를 평가 및 비교하였으며 결과는 표 2 와 같다. 실험 결과 Decision Tree 가 가장 높은 정확도와 F1 점수를 나타낸다는 것을 확인할 수 있다. 이러한 이유는 데이터 셋이 적어 자동으로 구성된 트리가 과적합되었고, 퍼셉트론의 경우 레이어가 충분히 넓거나 깊지 않아서라 판단하였다.

표 2. 기계학습 알고리즘 별 정확도 및 F1 Score.

Machine Learning Algorithm	Accuracy	F1 Score
SVM	88.5%	93.0%
KNN	92.7%	93.9%
Multilayer Perceptron	97.4%	97.7%
Random Forest	97.9%	98.7%
Decision Tree	99.2%	99.5%

Ⅳ. 결론 및 향후 연구

본 논문에서는 패킹된 악성코드에 대해 5 가지의 기계학습 알고리즘을 적용한 이진 분류 실험 및 비교하였다. 실험을 통해 본 논문에서 Decision Tree 가 가장 높은 정확도 및 F1 점수를 달성했으나, 이는 적은 데이터 셋으로 인한 현상으로 판단된다. 동시에 우리는 기존 방법인 시그니처 기반으로는 탐지할 수 없었던 커스텀 패킹 여부를 판단할 수 있었다. 향후 연구에서는 더 많은 악성코드를 수집하여 더 높은 정확도를 달성할 수 있도록 Autoencoder, Ensemble 또는 Multimodal 과 같은 다양한 방법을 적용 및 검증할 계획이다.

ACKNOWLEDGMENT

본 연구는 한국전력공사의 2018 년 착수 에너지 거점대학 클러스터 사업에 의해 지원되었음 (과제번호:R18XA05)

참 고 문 헌

[1] Cisco, "Defending against today's critical threats," 2019a(https://www.cisco.com/c/en/us/products/security/security-reports.html#~more-reports).

[2] Choi, Mi-Jung, et al. "All-in-One Framework for Detection, Unpacking, and Verification for Malware Analysis," *Security and Communication Networks*, vol. 2019, Article ID 5278137, 16 pages, 2019.

[3] Lyda, Robert, et al. "Using entropy analysis to find encrypted and packed malware," *IEEE Security & Privacy*, vol. 5, no. 2, pp. 40-45, Apr.2007.

강화학습 기반 링크가중치 조정 로드밸런싱 알고리즘 연구

임지윤°, 남석현 유재형, 홍원기
포항공과대학교 컴퓨터공학과

{limjiyoon, obiwan96, jhyoo78, jwkhong}@postech.ac.kr

A Study on the reinforcement learning based link weight adjustment load balancing algorithm

Jiyoon Lim°, Sukhyun Nam, Jae-Hyoung Yoo, James Won-Ki Hong
Department of Computer Science and Engineering, POSTECH

요 약

데이터센터 네트워크의 토폴로지와 트래픽 패턴은 기존의 인터넷과 다른 양상을 보인다. 특히 트래픽 패턴은 짧은 플로우 처리 완료 시간(flow completion time)과 높은 처리량(throughput)을 요구한다. 기존에 제안된 데이터센터 로드밸런싱 알고리즘의 경우 짧은 플로우에 대해 처리 능력이 부족하거나, 확장성에 한계가 있다. 본 연구는 이러한 기존 알고리즘의 문제점을 개선하기 위해 강화학습 기반 링크가중치 조정 알고리즘을 제안한다. 제안하는 알고리즘은 프로그래머블 스위치와 In-band 네트워크 텔레메트리 기법을 사용하여 높은 네트워크 가시성을 확보하고, 강화학습을 통해 모든 링크의 가중치를 조정 후 각 링크의 가중치에 따라 패킷을 전송함으로써 기존 로드밸런싱 알고리즘보다 빠른 플로우 처리 완료 시간을 갖을 것으로 기대한다.

Ⅰ. 서 론

데이터센터 내에서 동작하는 어플리케이션들은 네트워크 트래픽을 효과적으로 사용하기 위해 높은 양방향 대역폭 (bisection bandwidth)을 요구한다. 데이터센터 네트워크는 이러한 요구에 따라 기존의 인터넷과 다르게 각 호스트 간에 다중 경로를 제공하는 토폴로지를 사용하며[1], 트래픽은 기존의 인터넷과 다른 패턴을 보인다[2][3]. 즉, 10KB 이하의 짧은 플로우가 전체 플로우의 약 80%를 차지하며, 트래픽 양의 90%는 100MB 이상의 elephant 플로우에 의해 발생하고 있다. 이러한 추세에 따라 데이터센터 네트워크는 점점 더 빠른 플로우 처리 시간(flow completion time)과 높은 처리량(throughput)을 요구하고 있다. 이러한 요구사항을 만족하기 위해서는 데이터센터 네트워크의 특성을 반영하는 새로운 로드밸런싱 알고리즘이 필요하다. 기존에 주로 사용되어 온 로드밸런싱 알고리즘으로 ECMP (Equal-Cost Multipath)[3]가 있다. ECMP 는 패킷 헤더에 있는 플로우 관련 필드를 해싱하고 이를 바탕으로 트래픽을 각 최단 경로에 균등하게 분배하는 방식이다. ECMP 는 데이터센터 네트워크의 다중 경로를 활용하지만, 둘 이상의 elephant 플로우를 같은 경로로 보낼 경우 네트워크 혼잡이 발생하여 처리량이 매우 떨어진다는 한계점이 있다.

ECMP 의 한계점을 해결하기 위해 네트워크의 정보를 로드밸런싱에 활용하는 방법들이 제안되고 있다. 중앙 집중화된 컨트롤러에서 네트워크 전체의 정보를 활용하여 라우팅하는 방식[4], 각 스위치에서 네트워크 일부분의 정보만을 사용하여 라우팅하는 방식[5], 머신 러닝 기법을 사용하여 네트워크 상태를 학습하고 이를 사용하여 라우팅하는 방식[6][7]등의 연구가 진행되고 있다.

본 연구는 강화학습 기반 링크가중치 조정 로드밸런싱 알고리즘을 제안한다. 제안하는 알고리즘은 강화학습을 통해 네트워크 상태에 따라 모든 링크의 가중치를 학습한다. 또한 INT (In-band Network Telemetry)[8] 기법을 활용하여 높은 네트워크 가시성을 확보하고 이를 강화학습의 입력 값으로 활용한다. 제안하는 알고리즘은 모든 크기의 플로우에 대해 효과적으로 라우팅하여 빠른 플로우 처리 시간을 제공하고자 한다.

Ⅱ. 관련 연구

1. INT (In-band Network Telemetry)

정밀한 네트워크 정보를 모니터링 하기 위해 프로그래머블 스위치와 INT 기법을 활용하는 연구가 많이 진행되고 있다[8]. 프로그래머블 스위치는 P4[9] 라는 DSL (Domain-specific Language)로 헤더 포맷, 헤더 파싱 방법, match-action 테이블 구조 및 컨트롤 플로우를 프로그래밍 할 수 있다.

INT 는 프로그래머블 스위치를 활용하여 제어 평면의 개입없이 패킷 단위의 네트워크 정보를 모니터링하기 위해 설계된 프레임워크이다. INT 의 아키텍처 모델은 Source 스위치, Transit 스위치, Sink 스위치로 구성된다. Source 스위치는 패킷에 “telemetry instruction”이라고 불리는 헤더 필드를 삽입한다. Telemetry instruction 은 수집할 네트워크 정보를 정의하고, 수집된 정보를 어떻게 처리할지 결정한다. Transit 스위치에서는 telemetry instruction 에 의해 네트워크 디바이스에서 네트워크 정보를 추출하여 패킷에 삽입시키고, 패킷을 전송한다. Sink 스위치는 telemetry instruction 에 의해 수집된

네트워크 정보를 데이터베이스에 전송하고, 패킷에 삽입된 telemetry instruction 과 네트워크 정보를 제거한다. INT 로 수집할 수 있는 정보는 스위치 ID, 입출력 포트 번호, 패킷의 입출 시간, 홉 지연시간, 출력 포트에 대한 링크 이용률, 큐 점유율, 버퍼 점유율 등이 있다.

2. 강화학습

강화학습은 환경과 상호작용하여 피드백을 통해 학습하는 머신러닝 기법 중 하나이다. 그림 1 은 강화학습의 각 요소를 나타낸 것이다. 에이전트(Agent)는 환경을 관측하여 현재 상태(State)를 유추한다. 그리고 에이전트는 현재 상태에 대해서 자신만의 정책(Policy)을 통해 행동(Action)을 결정한다. 환경(Environment)은 에이전트의 행동에 영향을 받아 변화하고, 에이전트는 환경으로부터 행동에 따른 보상(Reward)을 얻게 된다. 강화학습의 목적은 최종적으로 얻는 보상의 총합을 최대화하는 것이다. 지도학습 (Supervised learning)과 비교하여 강화학습은 알파고와 같이 연관성이 있는 연속된 데이터에 대한 학습이나 레이블을 붙이기 어려운 데이터를 사용하는 경우에 대해 상대적 강점을 갖는다.

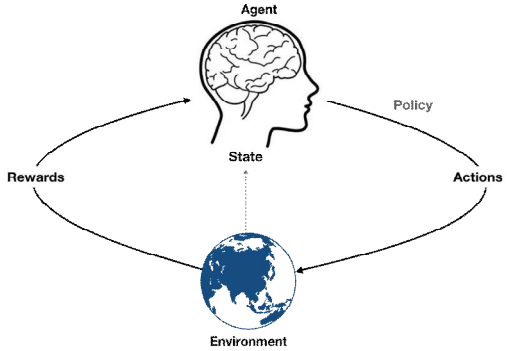


그림 1 강화학습의 요소

3. 로드밸런싱 알고리즘

중앙 집중화된 컨트롤러에서 모든 정보를 처리하는 로드밸런싱 알고리즘은 짧은 플로우에 대한 처리능력에 한계가 있다. 또한 네트워크의 일부분의 정보를 각 스위치에서 저장하고 스위치에서 라우팅하는 방식의 로드밸런싱 알고리즘은 네트워크 정보를 오버헤드가 크다는 문제가 있다. 다음은 이러한 문제를 해결하기 위해 제시된 머신러닝 기반 로드밸런싱 알고리즘들이다.

RILNET[7]은 로드밸런싱 알고리즘에 강화학습을 적용한 기법이다. RILNET 은 모든 edge 스위치 페어에 대해 플로우의 대역폭을 관측한다. 관측값은 합성곱 신경망(Convolution Neural Network)으로 상태를 축약한다. RILNET 의 action 은 모든 edge 스위치 페어에 대한 경로 가중치이다. 학습 알고리즘으로는 DDPG[10]를 사용하여 모든 edge 스위치 페어에 대해서 병목에 해당하는 링크 이용률의 평균값을 최소화하도록 학습시킨다. RILNET 은 강화학습에 대한 오버헤드가 크기 때문에 짧은 플로우에 대한 결정이 제대로 이루어지지 않는다는 한계가 있다.

HMMLB[8]는 은닉 마르코프 모형(Hidden Markov Model)을 로드밸런싱에 활용한다. 은닉 마르코프 모형을 통해 네트워크의 일부 스위치에서 추출한 링크 대역폭 데이터를 네트워크 전체 대역폭으로 추론한다. 그리고 추론한 네트워크 전체 대역폭을 기반으로 패킷 경로를 결정한다. HMMLB 는 Hedera[4] 대비 비슷한 성능을 유지하면서 시간 오버헤드를 유의미하게 감소시켰다.

본 연구는 INT 와 강화학습을 활용하는 로드밸런싱 알고리즘을 제안한다. 제안하는 알고리즘은 INT 를 사용하여 플로우에 대한 네트워크 정보를 수집하고, 순환 신경망[15]을 사용하여 연속적인 플로우 데이터로부터 네트워크 상태로 추론한다. 또한 추론한 네트워크 상태로부터 모든 링크에 대한 가중치를 출력한다. 제안하는 알고리즘은 학습에 사용하는 뉴럴 네트워크의 크기를 감소시킴으로써 더 빠른 라우팅이 가능하고, 짧은 플로우에 대해서도 경로를 결정할 수 있게 한다.

Ⅲ. 강화학습기반 알고리즘 제안

본 논문에서는 강화학습을 통해 각 스위치 별 출력 포트에 대한 가중치를 결정하는 로드밸런싱 알고리즘을 제안한다. 그림 2 는 제안하는 알고리즘 모델을 나타낸다. 알고리즘은 1) 네트워크 정보 수집 단계 2) 강화학습 단계로 이루어져 있다.

네트워크 수집 단계에서는 프로그래머블 스위치와 INT 기법을 활용하여 네트워크 정보를 수집하는 단계이다. 데이터평면에서는 프로그래머블 스위치에 P4 프로그램이 배포되어 다음과 같이 동작한다. 가장 먼저 각 스위치에 Source 스위치, Transit 스위치, Sink 스위치의 역할을 부여한다. Edge 스위치는 Source 스위치와 Sink 스위치의 역할을 한다. 패킷이 edge 스위치를 지날 경우 만약 해당 패킷에 telemetry instruction 이 삽입되어 있지 않다면 telemetry instruction 이 헤더 필드에 삽입된다. 만약 패킷에 telemetry instruction 이 삽입되어 있다면 삽입된 telemetry instruction 과 수집한 네트워크 정보를 추출하여 데이터베이스로 전송한다. aggregation 스위치와 core 스위치는 Transit 스위치가 되어 패킷이 해당 스위치를 지날 때마다 네트워크 디바이스로부터 네트워크 정보를 수집하여 패킷에 삽입한다. INT 기법으로 수집하는 네트워크 정보로는 각 스위치의 홉 지연시간, 버퍼 점유율, 링크 별 큐 점유율, 링크 이용률, 링크 대역폭 등이 있다.

강화학습 단계에서는 수집된 네트워크 정보를 바탕으로 모든 링크에 대한 가중치를 계산하고 이를 제어평면에 전달한다. 강화학습의 요소는 다음과 같이 정의한다. 에이전트는 데이터평면으로부터 수집한 INT 데이터를 관측한다. INT 데이터는 T 시간 동안 수집한 과거 K 개의 플로우 데이터를 말한다. 플로우 데이터는 각 링크의 사용량, 각 링크의 대역폭, 각 스위치의 버퍼 사용량, 각 스위치의 홉 지연시간을 사용한다. 네트워크 상태는 네트워크 전체의 링크 이용률, 각 링크의 대역폭, 각 스위치의 버퍼 사용량, 각 스위치의 홉 지연시간으로 정의한다. 제안하는 알고리즘은 연속된 데이터를 처리하는데 강점을 갖고 있는 순환 신경망을 사용하여 INT 데이터로부터 네트워크 상태를 추론한다. 강화학습의 행동은 네트워크 상태로부터 모든 링크에 대한 가중치를 결정하는 것으로 이는 순환신경망의 출력값에 해당한다. 각 링크의 가중치는 0 과 1 사이의 값이다. 강화학습의 보상으로는 평균 플로우 처리시간에 -1 을 곱한 값을 사용한다. 학습 알고리즘으로 DDPG 를 사용하여 보상을 최대화 하도록 학습시킨다.

DDPG 알고리즘은 Actor 네트워크, Target Actor 네트워크 Critic 네트워크, Target critic 네트워크, 총 네 개의 순환 신경망이 학습에 사용된다. Actor 네트워크는 행동을 결정하는 정책을 계산하는 네트워크이고 Critic 네트워크는 행동을 평가하는 Q-value 를 계산하는 네트워크이다. 나머지 두 Target 네트워크는 각각 Actor 네트워크와 Critic 네트워크를 복사한 네트워크로 학습

결과를 천천히 반영하여 학습에 안전성을 높이기 위해 사용된다. Actor 네트워크는 현재 상태와 Critic 네트워크에서 계산한 Q-value 를 사용하여 정책을 경사하강법(Gradient Descent)으로 학습한다. Critic 네트워크는 (현재 상태, 현재 행동, 보상, 다음 상태)로 이루어진 배열로 정의되는 transition 을 사용하여 손실함수(Loss function)을 최소화하도록 학습한다. 손실함수는 Actor 네트워크와 Critic 네트워크로 계산한 Q-value 와 Target Actor 네트워크와 Target Critic 네트워크로 계산한 Q-value 의 차이로 정의한다. 이후 각 Target 네트워크들의 값은 현재 Target 네트워크의 값과 해당하는 네트워크의 값을 일정 비율로 합친 값으로 업데이트한다.

강화학습 모듈은 INT 데이터가 들어오면 Actor 네트워크가 결정한 현재의 정책에 따라 행동을 결정하고 transition 을 버퍼에 저장한다. 이후 버퍼로부터 N 개의 transition 을 랜덤하게 선택하고 transition 으로부터 Critic 네트워크를 학습한다. 그리고 학습된 Critic 네트워크를 바탕으로 Actor 네트워크를 학습한다.

강화학습으로 결정된 모든 링크에 대한 가중치는 컨트롤러에 전달된다. 이후 플로우가 흐를 경우 각 스위치에서는 패킷 재배열 문제를 해결하기 위해 flowlet[11] 단위로 출력포트를 선택한다. 각 출력포트는 해당 스위치의 모든 출력포트 의 링크 가중치의 합에 대한 현재 출력포트의 링크 가중치의 비율로 선택된다. 이는 모든 경로로 패킷을 보내어 네트워크 전체의 정보가 전달될 수 있도록 하는 역할을 한다.

제안하는 알고리즘은 네트워크 일부분의 정보만을 사용하고 전체 링크의 가중치를 선택한다. 기존 강화학습기반 로드밸런싱 알고리즘에 비해 입력값과 출력값의 차원이 모두 작아 뉴럴 네트워크 크기 또한 작아지게 된다. 이는 강화학습에 의한 시간 오버헤드를 줄여 짧은 플로우에도 대응할 수 있도록 한다. 네트워크 전체 데이터를 사용하지 않고 일부만을 사용함에 따라 예상되는 성능이 저하되는 문제는 연속된 데이터에 맞는 순환 신경망을 사용하여 보완할 수 있다. 또한 주어진 네트워크 환경에서 데이터수집과 학습을 끊임없이 반복한다. 따라서 네트워크 환경이 바뀌어도 지속적인 학습을 통해 새로운 네트워크에 적응하여 기존 로드밸런싱 알고리즘과 비슷한 성능을 보여줄 수 있다.

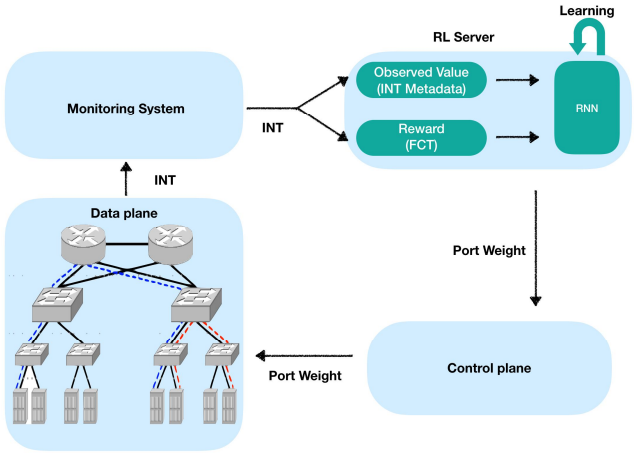


그림 2 강화학습기반 링크가중치 조정 알고리즘

IV. 결론

본 논문에서는 프로그래머블 스위치와 INT 기법을 사용해 네트워크 정보를 수집하고, 이를 활용한 강화학습 기법 알고리즘으로 링크가중치를 결정하는 로드밸런싱 알고리즘을 제안하였다. 제안하는 알고리즘은 네트워크 전체의 데이터가 아닌 연속된 플로우 데이터만을 사용하여 전체 네트워크 상태를 추측하고, action 의 수를 감소시켜 강화학습에 의해 발생하는 오버헤드를 최소화한다. 향후 연구로는 프로그래머블 스위치를 통해 실제 시스템을 구축하여, 데이터센터 네트워크 트래픽을 생성하여 기존 다른 알고리즘과 성능을 비교•분석할 예정이다. 마지막으로, 네트워크 크기에 따른 강화학습 알고리즘의 학습 속도와 패킷 처리시간을 고려하는 로드밸런싱 알고리즘을 개발한다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발)

참 고 문 헌

[1] Al-Fares et al: A scalable, commodity data center network architecture. In: ACM SIGCOMM Computer Communication Review, vol. 38, pp. 63- 74. ACM, 2008

[2] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, “Vl2: a scalable and flexible data center network,” in ACM SIGCOMM computer communication review, vol. 39, pp. 51- 62, ACM, 2009

[3] Benson, T., Akella, A., Maltz, D.A.: Network traffic characteristics of data centers in the wild. In: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, pp. 267- 280. ACM, 2010

[3] M. Chiesa, G. Kindler, and M. Schapira, “Traffic Engineering with Equal-Cost-MultiPath: An Algorithmic Perspective,”IEEE Conference on Computer Communications, 2014

[4] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, “Hedera: Dynamic Flow Scheduling for Data Center Networks,” NSDI, 2010

[5] M. Alizadeh et al., “CONGA: distributed congestion-aware load balancing for datacenters,”Proceedings of the 2014 ACM conference on SIGCOMM, 2014.

[6] Lin Q., Gong Z., Wang Q., Li J. “RILNET: A Reinforcement Learning Based Load Balancing Approach for Datacenter Networks,” In: Renault É., Mühlethaler P., Boumerdassi S. (eds) Machine Learning for Networking. MLN 2018. Lecture Notes in Computer Science, vol 11407. Springer, Cham, 2019

[7] Binjie He, Dong Zhang, Chang Zhao, Hidden Markov Model-based Load Balancing in Data Center Networks, The Computer Journal, , bxz142, https://doi.org/10.1093/comjnl/bxz142, 2019

[8] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, “In-band Network Telemetry via Programmable Dataplanes,” In ACM SIGCOMM, 2015.

[9] P. Bosshart et al., “P4: programming protocol-independent packet processors,” ACM SIGCOMM Computer Communication Review, vol. 44, no. 3, pp. 87-95, 2014.

[10] Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015

[11] E. Vanini, R. Pan, M. Alizadeh, P. Taheri, and T. Edsall, “Let it flow: Resilient asymmetric load balancing with flowlet switching,” in Proc. USENIX NSDI,Boston,MA, USA, 2017, pp.407-420

기계학습 기반의 VNF 최적 배치 모델에 관한 연구

박수현°, 홍지범, 유재형, 홍원기

포항공과대학교 컴퓨터공학과

{sh.park11, hosewq, jhyoo78, jwkhong}@postech.ac.kr

A Study on Machine Learning-based Optimal VNF Placement

Suhyun Park°, Jibum Hong, Jae-Hyoung Yoo, James Won-Ki Hong

Department of Computer Science and Engineering, POSTECH

요 약

NFV (Network Function Virtualization) 의 환경에서는 범용 서버에 소프트웨어로 구현되는 VNF (Virtualized Network Function)을 네트워크 토폴로지 상 적절한 위치의 노드에 배치하고 네트워크 상황에 따라 동적으로 관리함으로써, 다양한 트래픽 상황에 대해 신속하고 유연하게 대응할 수 있다. 하지만 비용과 서비스 품질 등을 고려하여 최적의 VNF 배치를 결정하고 적용하는 것은 복잡하고 어려운 문제이다. 특히, 결정된 배치를 실제 NFV 환경에 적용하는 데는 처리 시간이 소요되므로 배치가 적용되는 미래 시점의 상황을 예측하여 미리 결정하는 것이 필요하다. 본 논문에서는 MEC (Multi-access Edge Computing) 토폴로지에서 서비스 요청을 임의로 생성하여 ILP (Integer Linear Programming) 모델을 통해 시뮬레이션한 결과를 훈련 데이터로 사용하는 기계학습 모델을 도출한 후 테스트베드에서 성능을 검증하는 연구와 최종 모델을 적용한 VNF 배치 자동화 시스템을 제안한다. 이를 통해 VNF 의 배치를 효율적으로 관리할 수 있다.

Ⅰ. 서 론

인터넷에서는 점점 더 멀티미디어를 사용하는 다양한 형태의 high-throughput 서비스 요구가 증가하고 있다. 전통적인 네트워크 환경에서는 서비스 사업자(service providers, SPs)가 서비스 처리에 필요한 기능을 제공하는 미들 박스 (middle boxes) 장비를 네트워크 내부의 적절한 위치에 추가로 설치하고 이에 맞추어, 트래픽 흐름을 제어하는 등의 작업을 수행해왔다. NFV 기술은 이러한 고가의 미들박스들을 범용 서버에 가상화하여 구현한 VNF 들로 대체하고 실시간으로 변화하는 서비스 요구에 유연하게 처리하도록 하는 기술이다 [1].

NFV 기술은 동적으로 변화하는 네트워크 서비스 요청 (service request)에 대해 유연하게 대처할 수 있는 장점이 있지만, 이를 위해 비용과 서비스 품질을 고려하여 VNF 를 동적으로 배치하고 그에 따라 SFC (Service Function Chain)를 구성하는 문제는 네트워크 관리자가 수작업으로 수행하기 어려울 만큼 규모가 크고 복잡해진다. 이는 네트워크 토폴로지, 서버의 자원 할당, 네트워크 대역폭 할당, VNF 타입별 카탈로그와 각 시점의 서비스 요청 정보를 바탕으로 비용과 서비스 품질 등의 관점에서 최적의 해를 찾는 문제로 접근할 수 있다. 하지만 결정된 VNF 배치를 적용하는 데 처리 시간 (VNF deployment time)이 소요되기 때문에 VNF 배치 결정이 이루어지는 시점의 정보로 도출한 최적의 배치가 실제 적용되는 시점에는 최적이 아닐 수 있다는 문제가 있다 [2].

본 논문에서는 MEC 토폴로지에서 VNF 최적 배치를 결정하는 기계 학습 모델을 제안한다. 제안하는 방법은 서버의 자원 할당, 네트워크 대역폭 할당, VNF 타입별 카탈로그와 각 시점의 서비스 요청 정보를 바탕으로 최적의 VNF 배치를 산출하여 레이블링한 데이터를 생성한 후, 기계학습 기법을 통해 현재 시점의 정보를 입력으로 미래 시점의 최적 VNF 배치를 출력하는 모델을 학습시키는 것이다. 제안하는 모델의 성능이 검증되면, VNF 배치 결정을 자동으로 수행하는 시스템에 적용할 수 있다.

Ⅱ. 관련 연구

M. F. Bari [3]는 VNF 배치와 SFC 구성을 결정하는 문제를 서비스 수준 협약 (Service Level Agreement, SLA) 기준을 맞추는 범위에서 네트워크 운용 비용과 이용률을 최적화 시키는 문제로 정의하고, 이 문제에 대한 ILP 를 수학적 최적화 소프트웨어 패키지인 CPLEX 에서 구현하였다. 이때 네트워크 토폴로지, 서버의 자원 할당, 네트워크 대역폭 할당, VNF 타입별 카탈로그와 각 시점의 서비스 요청 정보를 활용하였다.

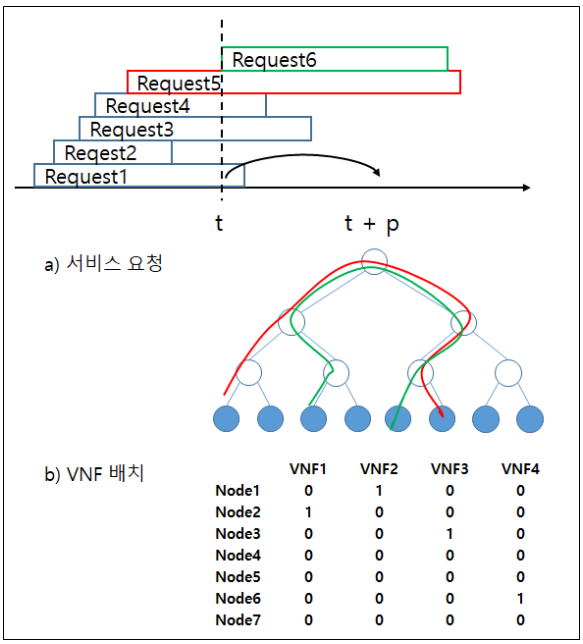
X. Zhang [2]은 VNF 배치에 대한 기존 연구들이 이미 도착한 서비스 요청에 대응하는 방식으로 접근하고 있음을 지적하며, 결정된 배치를 적용하는 처리 시간을 고려하여 서비스 요청을 도착 전에 미리 예측할 필요가 있다고 주장하였다. 이를 토대로 X. Zhang 은 VNF 배치를 선제적으로 계획하는 방법을 제시하였다. 마찬가지로 B. Li [1]도 미래의 VNF-SC (virtual network function service chain) 요청을 정확히 예측하기 위해 LSTM 기반의 딥러닝 (deep learning) 모델을 설계하였다.

S. Lange [4]는 시뮬레이션으로 구한 서비스 요청 데이터와 연구 [3]의 ILP 솔루션을 기반으로 미래 시점의 VNF 타입 별 가상 서버의 소요량을 예측하였다. 이 연구는 시뮬레이션 환경에서 서비스 요청 데이터를

생성하고, 이를 기반으로 생성한 데이터를 각 시점에서의 트래픽 데이터로 변환하여 ILP 솔루션의 입력 값으로 사용하였다. 또한 이 ILP 솔루션의 결과를 기반으로 각 시점의 VNF 타입 별 가상 서버 수량 증감에 대한 결정을 레이블링하였다. 이 데이터를 바탕으로 과거부터 현재 시점까지의 통계 정보를 입력 값으로 받아 예측 시간대의 VNF 타입 별 수량의 결정을 증가/유지/감소 결과 값으로 출력하는 기계학습 기반의 분류 모델을 제안하였다. 제안하는 모델은 60 초 이후의 시점에서 최적의 VNF 타입 별 수량 증감에 대해 ILP 결과 대비 75~80%의 정확도를 달성하였다.

Ⅲ. VNF 최적 배치 예측 모델

본 논문에서 제안하는 VNF 배치 예측의 형태는 그림 1 과 같다. 현재 시점 (t)에서 p 시간 이후 시점에 대한 최적의 VNF 배치를 예측하기 위해 두 가지 방법을 고려할 수 있다. 첫 번째 방법은, a)와 같이 미래 시점의 유효한 서비스 요청을 예측한 후, 이를 바탕으로 VNF 배치를 최적으로 계산한다. 두 번째 방법은, b)와 같이 새로운 서비스 요청이 도착하는 각 시점에 대해 서버 노드를 행으로 하고 VNF 타입을 열로 하는 VNF 최적 배치 행렬을 직접 예측하도록 한다. 본 연구에서는 두 번째 방법으로 학습을 진행하는 과정에 서비스 요청에 대한 예측이 내부적으로 포함될 수 있다고 판단하여, 두 번째 방법으로 진행한다.



	VNF1	VNF2	VNF3	VNF4
Node1	0	1	0	0
Node2	1	0	0	0
Node3	0	0	1	0
Node4	0	0	0	0
Node5	0	0	0	0
Node6	0	0	0	1
Node7	0	0	0	0

그림 1 VNF 최적 배치 예측 예시

지도 학습 방법으로 VNF 최적 배치 예측 모델을 훈련시키기 위해서, ILP 솔루션을 이용해 시간의 흐름에 따른 서비스 요청 정보를 생성하고, 각 시점에 대한 VNF 최적 배치 정답을 레이블링하는 과정은 다음과 같다. 먼저, MEC (Multi-access Edge Computing) 토폴로지 상에 출발지 노드, 목적지 노드, 도착시간, 지속시간, 트래픽, 최대 지연의 정보를 가지는 서비스 요청 정보를 생성한다. 이를 입력 값으로 하여 CPLEX 프로그램 기반의 ILP 솔루션 [3] 결과로 얻어지는 각 시점에서의 최적의 VNF 배치를 그림 1 의 b)와 같은 매트릭스 형태로 가공한다.

기계학습 모델의 입력 피처 (feature)로는 네트워크 토폴로지, 서버의 자원 할당, 네트워크 대역폭 할당, VNF 타입별 카탈로그와 같은 네트워크 환경 정보뿐만 아니라, 생성한 서비스 요청에 대한 ILP 솔루션의 결과인 VNF 배치 및 SFC 라우팅 정보를 바탕으로 각 노드의 자원 사용량, 대역폭 사용량 등을 계산하여 사용한다. 모델이 예측하는 VNF 배치 결과가 예측 대상이 되는 미래 시점의 ILP 솔루션의 결과와 같아지도록 목적 함수 (objective function)를 구성하여 ILP 솔루션의 VNF 배치 결과와 가까운 결과를 내도록 학습을 진행한다. 시뮬레이션을 통해 얻은 데이터 중 훈련 데이터에 포함되지 않은 테스트 데이터로 모델의 정확도를 평가한다.

다음은 전 단계에서 도출한 모델의 성능을 실제와 가까운 환경에서 검증하기 위해 시뮬레이션에서 가정한 토폴로지와 같은 테스트베드 (testbed)의 네트워크 환경을 구성한 후, 동일한 서비스 요청을 기준으로 트래픽을 생성한다. 시뮬레이션에서는 ILP 를 통해 생성하는 각 노드의 자원 사용량, 대역폭 사용량 정보를 테스트베드의 각 노드 서버에서 실제로 측정하여 입력 피처로 수집한다. 각 시점에 대해 VNF 최적 배치로 예측된 결과를 적용하였을 때와 ILP 솔루션의 결과로 도출된 VNF 배치를 적용하였을 때, VNF 처리 시간 (VNF processing time)과 패킷 누락 비율 (packet drop rate) 등의 QoS (Quality of Service)를 비교함으로써 예측 모델의 성능을 평가한다.

최종적으로 도출되는 VNF 배치 결정 모델은 현재 시점까지 이어지는 단위 구간 (time window)의 서비스 요청 정보와 그에 따라 변화하는 각 노드의 자원 사용량, 대역폭 사용량 등의 정보를 모니터링한다. 이를 바탕으로 5 분 이상 미리 VNF 최적 배치를 결정하여, QoS 성능이 ILP 솔루션 대비 80% 수준을 달성하는 것을 목표로 한다.

Ⅳ. 결론

본 논문에서는 MEC 토폴로지 상에서 네트워크의 다양한 상태 정보를 바탕으로 미래 시점의 VNF 최적 배치를 예측하는 기계학습 모델을 제안한다. 제안하는 모델은 기존 VNF 타입 별 수량 증감 결정에 대한 연구 [4]를 개선하여 VNF 타입 별 최적 배치를 네트워크 토폴로지 상 노드의 위치까지 구체적으로 예측한다. 최종적으로는 테스트베드에서 검증된 모델을 적용하여 VNF 배치를 최적으로 결정하는 시스템을 구축하는 것을 목표로 한다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부 (과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발).

참 고 문 헌

[1] B. Li, W. Lu, S. Liu, and Z. Zhu, “Deep-learning-assisted network orchestration for on-demand and cost-effective VNF service chaining in inter-DC elastic optical networks”, IEEE/OSA Journal of Optical Communications and Networking, vol. 10, no. 10, pp. D29-D41, Oct. 2018.

[2] X. Zhang, C. Wu, Z Li and F. C. M. Lau, "Proactive VNF Provisioning with Multi-timescale Cloud Resources: Fusing Online Learning and Online Optimization", in 2017 IEEE Conference on Computer Communications, pp. 1-9, Oct. 2017.

[3] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, “On Orchestrating Virtual Network Functions”, in 11th International Conference on Network and Service Management, pp. 50-56, Nov. 2015.

[4] S. Lange, et al., "Predicting VNF Deployment Decisions under Dynamically Changing Network Conditions", in 2019 15th International Conference on Network and Service Management, pp.1-9, Oct. 2019.

머신 러닝 기반의 비트코인 주소 분류 기법

이채현 O*, 고경찬*, 우종수**, 홍원기*

*포항공과대학교 컴퓨터공학과

**포항공과대학교 정보통신연구소

{chlee0211, kkc90, woojs, jwkhong}@postech.ac.kr

A Method of Machine Learning based Bitcoin Address Classification

Chaehyeon Lee O*, Kyunchan Ko*, Jongsoo Woo**, James Won-Ki Hong*

*Department of Computer Science and Engineering, POSTECH

**Graduation School of Information Technology, POSTECH

요 약

비트코인을 거래하기 위해서는 비트코인 주소가 필요한데, 이 주소는 실 사용자의 정보를 연결하지 않는다는 익명성을 띠고 있다. 이러한 익명성을 악용하여, 비트코인 네트워크에서 다양한 불법 거래들이 활발하게 일어나고 있고 피해가 막심하다. 이에 본 논문에서는 거래와 관련된 비트코인 주소의 특성을 파악하고 주소의 분류를 예측하는 방법론을 제안한다. 여러 서비스 (거래소, 마이닝 풀, 믹서, 겜블링, 암거래 시장 - 실크로드)에 활용된 트랜잭션을 카테고리 별로 수집하고, 수집된 트랜잭션으로부터 연관된 비트코인 주소와 80 개의 특징을 추출한다. 그리고 머신 러닝 모델을 이용해 특정 비트코인 주소가 어떤 카테고리에 속하는지 분류해보았고 최고 약 84%의 분류 정확도를 가짐을 확인했다.

I. 서 론

비트코인 [1]은 P2P 네트워크 구조의 탈 중앙화 시스템으로, 제 3 자의 개입 없이 암호 화폐의 거래가 가능하다. 참여자가 동일한 데이터를 유지함으로써 투명한 거래가 가능하며 데이터의 위/변조가 불가능하다는 특징이 있어 크게 주목받고 있다. 비트코인을 거래하기 위해서는 비트코인을 전송 받을 비트코인 주소가 필요하며, 한 명의 유저 또는 entity 는 여러 개의 주소를 소유할 수 있다.

그러나 이러한 비트코인 주소는 실 소유주의 현실 정보와 연결되지 않는 익명성을 띠고 있다. 이러한 익명성을 악용하여, 비트코인 네트워크에서 다양한 불법 거래들이 활발하게 일어나고 있고 피해가 막심하다 [2, 3]. 대표적인 암거래 사이트인 실크로드를 통해 불법 무기, 불법 마약, 도난품, 악성 코드 등이 비트코인을 이용해 거래되었으며 그 거래 금액은 사이트가 폐쇄되기 전까지 약 707M 달러에 달한다 [4]. 실제로 비트코인 네트워크 상에서는 다크넷을 통한 불법 물품 거래 뿐 아니라, 자금 세탁, 스캐밍과 같은 불법적인 활동들이 꾸준히 일어나고 있으며, 이러한 활동들은 암호 화폐 관련 법 제정을 저해하는 요소로 작용해왔다. 따라서 네트워크 상에서 일어나는 불법 거래들을 사전에 탐지할 수 있는 시스템이 필요하다. 비트코인, 이더리움 [5], 모네로 [6] 등은 다크넷 상에서의 불법 거래에 많이 사용되어 왔지만, 본 연구에서는 다크넷에서 가장 많이 사용된 비트코인에 초점을 맞춘다. 악성 유저들은 반복해서 불법 거래를 발생시키며, 그 과정에서 한 명의 유저는 여러 개의 비트코인 주소를 활용한다. 따라서 본 연구에서는 불법 거래를 탐지하기에 앞서, 특정 비트코인 주소들이 어떠한 목적으로, 어떠한 서비스를 위해 사용되었는지를 분류해보았다. 비트코인 거래와 관련한 비트코인 주소와 특징들을 추출하고, 지도 학습 기반의 머신 러닝 분류 모델 [7]을 이용해 특징들을 학습시킴으로써 비트코인 주소가 어느 클래스에 속하는지 파악해보았다.

II. 관련 연구

Toyota [8]는 트랜잭션 패턴을 분석하여 HYIP (High Yielding Investment Program)인지 아닌지를 식별하는 연구를 진행하였다. 수동적으로 HYIP 와 non-HYIP 주소를 식별하고 비트코인 주소와 연관된 트랜잭션의 수, 채굴된 블록의 수 등의 특징을 추출하였다. 비트코인 주소를 HYIP 또는 non-HYIP 로 라벨링 하여 supervised 학습을 통해 사이버범죄 집단을 분류하였다. 83%의 분류 정확도를 보여주었다.

Kanemura [9]는 다크넷 [10]과 관련된 비트코인 트랜잭션과 주소들을 분석하였고 하나의 Entity 에 속하는 여러 개의 주소들의 라벨을 결정하기 위해 voting-based 시스템을 제안했다. 다크넷 마켓과 관련한 73 개의 특징들을 추출하고 supervised 분류기를 이용해 데이터를 학습시킨 후, 특정 주소가 DNM 에 속하는 주소인지 non-DNM 에 속하는 주소인지 분류를 결정하였다. 논문에서 제안한 Majority voting 기반의 voting method 를 통해 81%에 해당하는 분류 정확도를 도출했다.

본 논문의 저자는 이전 연구를 통해, 비트코인 트랜잭션의 특징만을 이용해 불법 거래를 탐지하는 연구를 진행했다 [11]. 불법 활동과 연관된 비트코인 주소나 비

트코인 클러스터를 분류하는 연구는 사전에 여러 차례 진행되어 왔지만 트랜잭션의 특징만을 이용해 불법 거래를 탐지한 사전 연구를 찾을 수 없었기 때문에 트랜잭션으로부터 9 개의 특징을 추출하고, 하나의 라벨을 붙여 10 개의 특징을 이용해 머신러닝 분류 모델을 학습시켰다. 실험 결과 0.9 에 달하는 F1 score 를 얻을 수 있었지만 테스트 셋이 overfitting 되었을 가능성이 있고, 트랜잭션의 불법 여부를 결정짓기에 특징의 수가 너무 적다고 판단하였다.

따라서 이전 연구에 이어, 비트코인 트랜잭션의 특징에서 비트코인 주소의 특징으로 범위를 확장하였고, 특징의 수를 늘려 어떤 특징들이 분류 모델에 영향을 끼치는지 확인해보았다.

III. 비트코인 주소 분류 방법론

비트코인 주소의 카테고리를 분류하기 활용한 4 단계의 방법론을 소개한다. 1. 트랜잭션 수집, 2. 비트코인 주소 및 특징 추출, 3. 기계학습 분류 모델 학습, 4. 실험 및 검증 단계가 이에 해당하며 그림 1 과 같은 절차를 따른다.

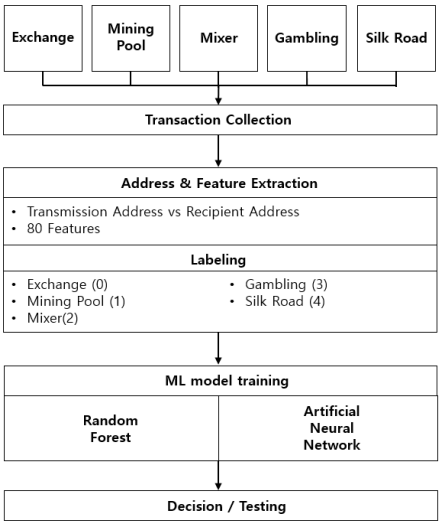


그림 1 불법거래 탐지 절차

1. 트랜잭션 수집

기계학습 모델을 구현하기에 앞서, 공개되어있는 포럼 사이트로부터 트랜잭션의 해시 리스트를 수집한다. WalletExplorer.com [12]은 트랜잭션의 해시를 카테고리 별(거래소, 마이닝 풀, 서비스 등)로 공개하고 있다. 본 논문에서는 믹서, 거래소, 겜블링, 마이닝 풀, 실크로드의 총 5 개 카테고리로부터 데이터를 수집하였다. 각 카테고리 별로 2016 년 1 월 1 일부터 2019 년 10 월 16 일까지의 데이터를 수집하였고, 실크로드의 경우는 사이트가 폐쇄되기 전까지의 데이터만을 포함한다.

2. 비트코인 주소 및 특징 추출

수집된 트랜잭션 해시로부터 하나 이상의 전송 주소와 하나 이상의 수신 주소를 추출할 수 있다. 수집한 트랜잭션 해시를 인자로 하여 JSON-RPC [13]를 요청한 결과 트랜잭션의 디테일한 정보를 비트코인 클라이언트 [14]로부터 얻어올 수 있다. 특정 비트코인 주소는 오직 전송 주소로만 사용되거나 수신 주소로만 사용되며, 전송과 수신을 위해 모두 사용되는 경우도 있다. 아래의 표 1 과 그림 2 는 카테고리 별 수집된 트랜잭션의 수와 추출한 수신/송신/총 비트코인 주소의 분포를 보여준다. 마이닝 풀과 믹서의 경우 상대적으로 적은 비율의 주소들이 추출되었고, 이를 통해 특정 주소가 해당 서비스를 위해 트랜잭션에 여러 번 등장했음을 유추해 볼 수 있다.

비트코인 주소를 추출한 후, 불법 거래들이 띠는 공통적인 패턴을 분석하기 위해 주소들로부터 특징을 추출한다. 28 개의 특징을 선정하였고, 일부 특징들은 평균값, 총합, 최소값, 최대값의 4 개의 값들을 포함한다. 특정 비트코인 주소는 우리가 수집한 트랜잭션에 여러 번 등장할 수 있으므로 동일한 비트코인 주소가 발견될 시 해당 값들을 업데이트한다. 전송 주소와 수신 주소 별로 특징이 추출되며, 0 의 값이 중복되는 것을 방지하기 위해 전송 주소의 경우엔 비트코인 수신과 관련한 필드들이 −1 로, 수신 주소의 경우엔 전송과 관련한 필드들이 −1 로 세팅된다. 다시 말해, 추출된 특징의 모든 값들이 −1 을 포함하지 않는다면 해당 주소는 전송과 수신 모두를 위해 사용된 주소이다. 4 개의 값(평균값, 총합, 최소값, 최대값)을 갖는 일부 특징들을 포함해 총 80 개의 특징을 추출하였고 각 카테고리 별로 0 에서 4 까지의 라벨을 부여하였다. 표 2 와 표 3 은 카테고리 별 라벨값과 추출한 특징에 대한 설명을 명시한다.

표 1 카테고리 별 추출된 거래 및 주소의 분포

Category	Transmission addresses	Recipient addresses	Total addresses	Transactions
Exchange	1,395,325	6,736,265	8,665,943	761,494
Mining Pool	218,476	1,036,143	1,375,327	325,800
Mixer	178,721	480,754	718,915	93,200
Gambling	726,210	3,960,029	5,345,783	752,300
Darknet (Silk Road)	704,376	938,730	2,305,872	956,186

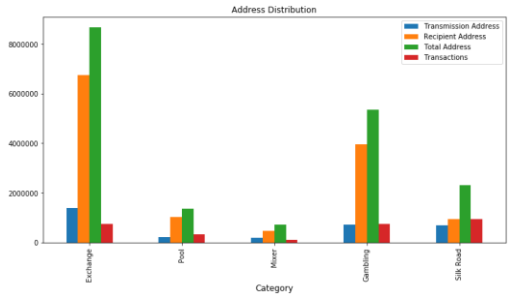


그림 2 카테고리 별 데이터 셋의 분포 비교

표 2 카테고리 별 라벨

Category	Label
Exchange	0
Mining Pool	1
Mixer	2
Gambling	3
Darknet (Silk Road)	4

3. 기계학습 분류모델 학습

수집된 주소들의 카테고리 분류를 위해 Random forest [15]와 DNN [16]두 가지의 기계학습 모델을 사용한다. 두 모델은 sklearn [17]과 tensorflow [18]를 이용해 구현하였으며 DNN 모델은 80 개의 특징을 학습시키기 위해 50 개의 노드를 가진 하나의 히든 레이어를 추가하였다.

표 3 추출된 특징과 Description

Name	Description	Etc
Bitcoin amount (transmit/receive)	Transmitted/Received bitcoin amount	
Total bitcoin amount (transmit/receive)	The amount of total bitcoin transmitted/received by the transaction associated with the transmission address	avg
Transaction fee (transmit/receive)	Transaction fees associated with bitcoin transmission/reception	sum
Sibling inputs/outputs (transmit/receive)	The number of sibling inputs/outputs	min
Sibling inputs/outputs_out/in (receive/transmit)	The number of outputs/inputs associated with bitcoin transmission/reception	max
Unique address (transmit/receive)	The number of unique transmission/receiving addresses	
Unique address_out/in (receive/transmit)	The number of unique receiving/transmission addresses associated with bitcoin transmission/reception	
Transaction Size (transmit/receive)	Transaction size associated with bitcoin transmission/reception	
Block Interval (transmit/receive)	The interval of the blocks related to the transmission/reception transaction	
Relevant transaction number (transmit/receive)	The number of transactions associated with the transmission/receiving address	
Lifetime (transmit/receive)	Life time of the transmission/receiving address	
First block (transmit/receive)	Block height where the transmission/receiving address first appeared	
Total transaction number	Total number of transactions associated with the address	
Total life time	Lifetime of the address	
Label	Classification of the address	

4. 모델 학습 및 테스트

학습이 완료되고 나면, test set 을 이용해 기계학습 모델이 제대로 구현되었는지 검증한다(그림 3). 구현이 잘 되었다면, 해당 모델을 활용해 주어진 특정 비트코인 주소가 어느 카테고리를 위해 사용되었는지 분류할 수 있다. 각 주소의 실제 라벨값과 예측값을 비교하여 모델이 얼마나 높은 정확도를 가지는지 확인할 수 있다.

lifetime_recv	lifetime_total	init_trns_block	init_recv_block	curr_trns_block	curr_recv_block	label	predicted
28610	28610	-1	577060	-1	577060	0	0
81636	61	520045	519984	520045	519984	2	2
-1	349145	256237	-1	256237	-1	4	4
371568	76	233367	233291	233367	233291	4	4
26277	26277	-1	466699	-1	492976	2	2
153551	7747	456638	448891	456638	448891	3	3
143460	143460	-1	458148	-1	458148	2	2
-1	380784	224025	-1	224025	-1	4	4
-1	46194	558509	-1	558509	-1	0	3
169877	169877	-1	432065	-1	432065	1	3
70557	70557	-1	531063	-1	531063	2	2
3716	3716	-1	548163	-1	551879	2	0
157921	157921	-1	444491	-1	444491	3	3
-1	33363	572057	-1	572057	-1	0	2
10887	10887	-1	595312	-1	595312	0	0
-1	378697	226127	-1	226127	-1	4	4
21590	21590	-1	584357	-1	584357	0	0
4763	4763	-1	438068	-1	442831	3	3
114093	114093	-1	493579	-1	493579	0	0
192875	192875	-1	408968	-1	408968	1	1

그림 3 테스트 셋의 예측 결과

IV. 실험 및 결과

그림 4 는 카테고리 별 추출된 트랜잭션과 주소의 분포를 나타낸다. 추출한 총 주소의 수는 18M 에 달하기 때문에 하드웨어 자원의 한계로 모든 데이터를 학습해 활용할 수 없다. 또한 각 카테고리 별로 불균형한 데이터 분포를 이루므로 실험을 위해서 데이터 셋을 카테고리 별로 랜덤하게 선정한다. 데이터 셋의 크기를 달리하여 여러 번 실험을 진행하며, 총 데이터 셋을 60:40 의 비율로 학습 데이터와 테스트 데이터로 나누었다.

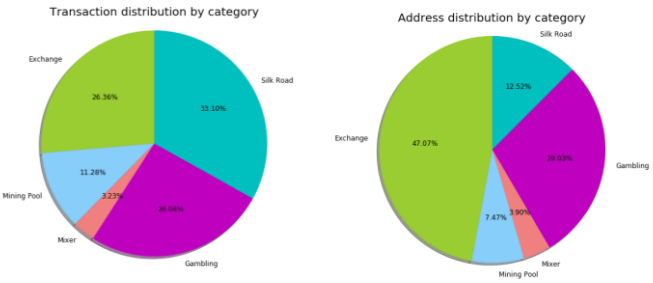


그림 4 카테고리 별 트랜잭션과 주소의 분포 비율

1. 특징 중요도

첫 번째 실험에서는 학습에 사용된 80 개의 특징들 중 분류 성능에 가장 영향을 많이 미치는 특징이 무엇인지 알아보았다. 각 카테고리 별로 2,000 개의 데이터를 랜덤하게 선택해 총 10,000 개의 데이터 셋을 이용하였고, 가장 중요도가 높은 Top 20 개의 특징을 조사하였다(그림 5). 실험 결과, 비트코인 수신과 관련한 특징들의 중요도가 높으며, 이는 데이터 셋에 수신 관련 주소가 많이 포함되었기 때문이라고 예측해볼 수 있다. 가장 영향을 많이 미친다고 판단되는 요소는 Lifetime 으로, 수집된 트랜잭션 셋에 걸쳐 특정 주소가 얼마나 오랫동안 해당 서비스를 위해 사용되었는지를 나타내는 지표이다. 특정 서비스가 동일한 주소를 반복해서 사용하는 경우 상대적으로 긴 Lifetime 을 가진다. 이 외에도 비트코인 주소가 수신한 총 비트코인의 양, 트랜잭션의 사이즈와 트랜잭션 수수료가 분류 정확도에 큰 영향을 미친다.

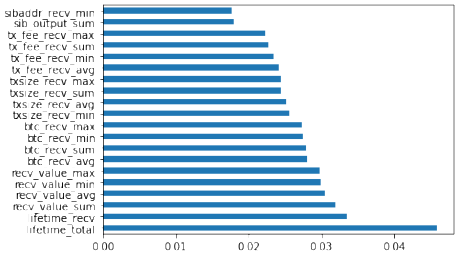


그림 5 카테고리 별 트랜잭션과 주소의 분포 비율

2. 분류 성능 비교

Random Forest Classifier 분류 성능 비교를 위해 각 카테고리 별로 1,000 개에서 200,00 개의 데이터를 랜덤하게 선택하여 여러 번의 실험을 반복하였다. 데이터 셋의 크기가 커질수록 분류 정확도가 높게 측정되었으며 최고 0.84 의 정확도를 보여주었다(표 4). 표 5 의 precision, recall, F1 score 는 카테고리 별로 분류가 얼마나 잘 되었는지 나타내는 지표이다. 실크로드와 관련된 주소의 경우, 모든 지표가 1.0 의 값을 가졌고 이는 실크로드와 관련된 주소들이 에러 없이 잘 분류되었음을 보여준다.

DNN 각 카테고리 별로 10,000 개에서 30,000 개의 데이터를 랜덤하게 선택하여 총 세 번의 실험을 반복하였다. 실험 결과 Random forest classifier 보다 상대적으

로 낮은 정확도와 F1 score 를 도출했다. 데이터 셋의 크기와 무관한 실험 결과를 보였으며 최고 정확도는 64% 이다(표 6). 믹서와 겜블링 관련 주소의 경우 단지 50% 정도만을 올바르게 분류하고 있음을 알 수 있었고, 그에 반해 실크로드 관련 주소는 Random forest classifier 와 같이 거의 정확히 분류해 줌을 확인해보았다(표 7).

표 4 Random forest classifier 의 정확도

Each data set	Accuracy
1,000	0.741
3,000	0.782
5,000	0.789
10,000	0.804
30,000	0.825
50,000	0.833
100,000	0.838
200,000	0.844

표 5 Random forest classifier 의 카테고리 별 정확도

Category	Precision	Recall	F1-Score
Exchange	0.82	0.74	0.78
Mining Pool	0.86	0.85	0.86
Mixer	0.78	0.86	0.82
Gambling	0.77	0.77	0.77
Silk Road	1.0	1.0	1.0

표 6 DNN 의 정확도

Each data set	Accuracy
10,000	0.646
20,000	0.620
30,000	0.614

표 7 DNN 의 카테고리 별 정확도

Category	Precision	Recall	F1-Score
Exchange	0.62	0.52	0.56
Mining Pool	0.77	0.56	0.65
Mixer	0.45	0.45	0.45
Gambling	0.51	0.46	0.48
Silk Road	0.99	0.98	0.99

V. 결론 및 향후 연구

본 연구에서는 머신 러닝 기반의 분류 모델을 이용해 여러 카테고리의 비트코인 주소들을 분류해보았다. 거래소, 마이닝 풀, 믹서, 겜블링, 암거래 시장(실크로드)에 활용된 트랜잭션을 카테고리 별로 수집하고, 수집된 트랜잭션으로부터 연관된 비트코인 주소와 80 개의 특징을 추출한다. 그리고 머신 러닝 모델을 이용해 특정 비트코인 주소가 어떤 카테고리에 속하는지 분류해보았고 최고 약 84%의 분류 정확도를 가짐을 확인했다. Random forest classifier 가 DNN 에 비해 상대적으로 높은 정확도를 보여주었고, 두 모델 모두 실크로드와 관련된 주소를 잘 분류해줌을 확인했다.

본 연구는 여러 개선사항들이 존재한다. DNN 성능 개선을 위해 다양한 머신 러닝 기법을 적용해 볼 예정이다. 히든 레이어의 수와 노드의 수를 조정하거나 80 개의 특징들 중 중요도가 높은 특징만을 이용해 재실험해 본다. 또한 다른 딥러닝 모델을 활용해 분류 정확도를 개선할 수 있다. 두 번째로, 특정 트랜잭션과 연관된 비트코인 주소들의 분류를 예측한 결과를 바탕으로 Majority voting 을 적용한다면, 더 나아가 해당 트랜잭션이 어떤 카테고리에 속하는지 판단할 수 있다. 실험 결과, 암거래 시장과 관련된 비트코인 주소가 거의 정확히 분류됨을 확인할 수 있었고, 이는 추후 트랜잭션의 불법 여부를 판단할 수 있는 가능성을 보여준다

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 임 (No.2018-0-00539)

참 고 문 헌

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).
[2] Harvey, Campbell R. "Bitcoin myths and facts." (2014).
[3] Bitcoin Magazine. Bitcoin magazine: Bitcoin news, bitcoin charts, events. Available at https://bitcoi**nmagazine.com/articles/darknet-markets-cant-live-with-or-without-bitcoin.**
[4] Wikipedia. Silk road (marketplace). Available at https://en.wikipedia.o*rg*/wiki/Silk_Road_(**marketplace).**
[5] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow pape*r, 151(2014):1–32, 2014.
[6] monero. Zero to monero. Technical report, 2018. Available at https: **//www.getmonero.org/library/Zero-to-Monero**-1-0-0.pdf
[7] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learn ing: *A* review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.
[8] K. Toyoda, T. Ohtsuki and P. T. Mathiopoulos, "Identification of High Yielding Investment Programs in Bitcoin via Transactions Pattern Analysis," *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Singapore, 2017, pp. 1-6.
[9] Kota Kanemura, Kentaroh Toyoda, and Tomoaki Ohtsuki. Identification of darknet markets' bitcoin addresses by voting per-address classification results. In *2019 IEEE International Conference on Blockchain and Cryp tocurrency (ICBC)*, pages 154–158. IEEE, 2019.
[10] Wikipedia. Darknet market. Available at https*://en*.m.wi**kipedia.org/wiki/Darknet_market.**
[11] LEE, Chaehyeon, et al. Toward Detecting Illegal Transactions on Bitcoin Using Machine-Learning Methods. In: *International Conference on Blockchain and Trustworthy Systems*. Springer, Singapore, 2019. p. 520-533.
[12] Walletexplorer: smart bitcoin block explorer. Available at **https:*//www*.walletexplorer**.com*/*.
[13] Bitcoin.org. Bitcoin core json apis. Available at https://bitcoin.org/e*n/***developer-reference#bitcoin-core-apis.**
[14] Bitcoin core. Available at https://bitcoin.o*rg/en/***bitcoin-core/.**
[15] Mahesh Pal. Random forest classifier for remote sensing classification. *In ternational Journal of Remote Sensing*, 26(1):217–222, 2005.
[16] Jacek M Zurada. *Introduction to artificial neural systems*, volume 8. West publishing company St. Paul, 1992.
[17] scikit-learn: Machine learning in python. Available at **https://scikit-learn.org/.**
[18] tensorflow: An end-to-end open source machine learning platform. Available at https://w**ww.tensorflow.org/?hl=en.**

이더리움 네트워크 노드 탐색 및 노드 영향력 분석

맹수훈*, 주홍택
*계명대학교, 계명대학교
*maengsoohoon@gmail.com, juht@kmu.ac.kr

Ethereum Network Node Discovery and Node Influence Analysis

Soohoon Maeng*, Hongtaek Ju
*Keimyung Univ., Keimyung Univ.

요 약

이더리움은 영구적으로 데이터를 기록하는 탈중앙화시스템으로서 비트코인과 더불어 대표적인 블록체인 네트워크이다. 블록체인 네트워크에서는 이중 지불 문제, DDos 문제, 51%공격, 시빌 공격과 같은 공격이 계속 되고 있고 이에 대한 해결 방법은 노드의 숫자와 노드의 영향력과 밀접한 관계가 있다. 따라서 본 논문에서는 이더리움 클라이언트(Ethereum Client)를 탐색노드로 사용하여 이더리움 네트워크에 참여하고 있는 노드를 탐색하고 이 노드들의 영향력을 분석한 결과를 제시한다. 탐지 노드는 이더리움 네트워크에 참여하고 있는 노드로 파악된 피어(Peer)를 버켓(Bucket)에 보관하는데 이 피어를 많이 보관하기 위한 방법을 제시한다. 또한 이더리움 네트워크에 참여하고 있는 노드 탐색과정을 분석하여 노드 영향력을 분석하였다. 실험을 통해 하루 동안 30만개의 노드를 탐색하였고 이들 간의 피어 관계 정보를 수집하여 이더리움 네트워크에 참여한 노드 중에서 영향력이 큰 139개의 노드를 확인했다. 또한 국가별 노드 운영 수치를 시각화하고 영향력이 큰 노드에 대한 스냅샷을 제공한다. 연구의 결과는 영향력이 큰 노드(Node)를 중심으로 악의적인 행위를 하는 보안 공격 탐지 방법, 그 외에 노드의 영향력과 이더리움 네트워크 성능과의 관계를 파악하고 특정 노드에서 생성된 블록이 전파되는 경로를 추적하는 연구의 기초 자료로 활용될 수 있다.

Ⅰ. 서 론

블록체인(Blockchain)은 합의 알고리즘(Consensus Algorithm)을 통해 위변조 없이 영구적으로 데이터(Data)를 기록하는 탈중앙화시스템(Decentralized System)으로 사토시 나카모토(Satoshi Nakamoto)에 의해 처음 제안되었다[1]. 이더리움(Ethereum)은 2015년 비탈린 부테릭(Vitalik Buterin)에 의해 가상 머신인 EVM(Ethereum Virtual Machine)과 스마트 컨트랙트(Smart Contract)기술을 접목시켜 구현된 탈중앙화 플랫폼이다[2]. P2P(Peer-to-Peer) 네트워크를 통해 연결된 이더리움 노드(Node)는 발생한 트랜잭션(Transaction)을 일정 주기로 블록(Block)에 기록되고 이더리움 네트워크에 공유되어 위변조 없이 영구적으로 기록된다.

이더리움 네트워크의 노드 탐색(Node Discovery)과정은 Kademlia와 RLPx(Recursive Length Prefix eXtension)를 기반으로 구현된다[3]. 이더리움은 비트코인과 함께 블록체인을 대표하는 탈중앙화시스템으로 현재까지도 마이닝 독점, 이중 지불 문제, DDos 공격, 51%공격, 시빌 공격과 같은 보안 문제가 발생하고 있다[4, 5, 6, 7]. 이러한 문제의 해결 방안은 노드의 숫자와 그 노드의 영향력을 바탕으로 도출되어야 한다. 또한, 비트코인과 달리 이더리움 네트워크의 불균형적인 영향력 분석에 대한 연구는 이루어지지 않았다.

본 논문에서는 전 세계에 분포한 이더리움 네트워크의 노드를 RLPx 프로토콜을 중심으로 탐색하는 방법을 제시한다. 또한 탐색한 노드의 피어 관계를 수집하여 탐색한 노드의 영향력을 분석한 결과를 제시한다. 분석된 결과는 향후 이더리움 네트워크 성능에 미치는 영향을 파악하고, 악의적인 노드에 대한 모니터링 방법과 더불어 이더리움 블록(Block)의 전파되는 과정을 추적 하는 연구에 대한 논의의 단초를 제시한다.

본 논문의 2장은 관련 연구로 비트코인 네트워크에서의 영향력 있는 노드 분석 방법을 소개하고 이더리움 P2P 네트워크에 대하여 설명한다. 3장에서는 노드 탐색 방법과 실험 과정을 설명하고 4장에서는 실험과 분석 결과를 제시하고 5장은 본 논문의 결론이다.

Ⅱ. 관련 연구

2.1 비트코인 네트워크의 노드 영향력 분석

Johnson 외 5명은 비트코인 네트워크에서 블록과 트랜잭션의 전파 지연을 줄이기 위해 많은 피어와 연결된 노드를 탐색하였고, 비트코인 네트워크에 직접 참여하지 않고 영향력을 발휘하는 노드가 존재함을 밝혔으며 이는 DDos 공격자들의 표적이 될 수 있다는 결과를 도출 했다[8]. Andrew Miller외 7명은 비트코인 네트워크에는 불균형적으로 영향력을 가진 노드가 존재하며 전체 2%의 노드가 전체 마이닝(Mining)의 3/4를 차지하는 결과를 도출했다[9]. 이와 같이 비트코인 네트워크에서 노드의 영향력에 대한 분석이 이루어졌으나 이더리움 네트워크에서의 노드 영향력에 대한 분석은 이루어지지 않았다.

2.2 이더리움 P2P 네트워크

이더리움 클라이언트(Ethereum Client)는 실행 초기에 부트스트랩 노드(Bootstrap Node)에게 노드 목록을 요청하고 응답 메시지를 통해 노드 목록을 전달 받는다[10]. 전달 받은 메시지에 포함된 노드 정보를 버켓(Bucket)에 저장한다. 이후에 RLPx와 Ethereum Wire Protocol을 구동하

여 이더리움 네트워크에 참여하고 블록 동기화 과정을 수행 한다. 이더리움 P2P 네트워크를 구성에 사용되는 RLPx 프로토콜과 Ethereum Wire 프로토콜은 그림 1과 같이 수행한다.

RLPx는 Kademlia알고리즘을 기반으로 이더리움의 인코딩방식과 암호화를 추가하여 P2P 비구조화 오버레이 네트워크를 구성한다. RLPx는 기본적으로 공개키를 노드ID로 사용하고 각 노드의 논리적 거리(Distance)는 두 노드 ID의 XOR연산 값을 사용한다[11]. 이더리움 네트워크 참여 인증은 타원 곡선 공유 대칭 키인 ESIES을 교환하여 TCP연결을 생성한다. 이더리움 클라이언트는 RLPx 프로토콜 수행을 통해 논리적 거리 별 피어 정보를 버켓(Bucket)에 수집하고 버켓의 크기는 16개이다. 이더리움 네트워크 참여 요청 메시지는 500ms의 타임아웃 내에 응답하고 RLPx 프로토콜 동작은 5s이내에 완료한다[9]. 이더리움의 버켓은 테이블(Table)의 타이머(Timer)를 통해 주기적으로 갱신한다. 버켓은 피어 활성화 여부를 Ping메시지를 통해 30초 주기로 확인하고, Ping에 대한 응답이 없을 경우 버켓에서 삭제하며 피어 목록은 활성화 상태인 노드로 목록을 갱신한다. 그림 1에서 RLPx 부분의 마지막 단계인 Find node 메시지는 Ping메시지 이후 교환되며 피어와 연결된 이웃피어(Neighbor Peer)의 결과 값을 IP형태로 응답받는다. 이웃피어는 피어와 연결되어 이더리움 네트워크에 참여한 제3의 노드이다. Find Node 메시지를 보낸 노드는 응답으로 받은 이웃피어들을 잠재적 피어가 될 가능성이 있는 후보로 간주하여 피어목록을 보관하는 버켓에 추가한다. 이더리움 네트워크에 참여하고 있는 노드는 Ping메시지와 Findnode를 이용하여 피어목록을 주기적으로 갱신하고 이웃피어를 탐색한다.

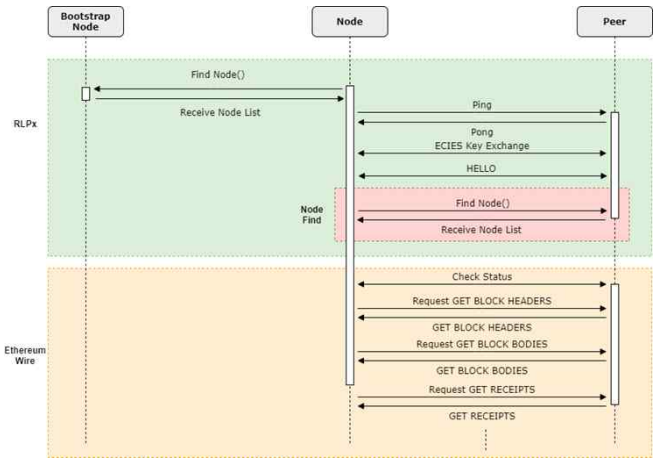


그림 1. Ethereum Network Protocols

Ethereum Wire 프로토콜은 이더리움 네트워크에 최초로 참여한 노드가 RLPx 프로토콜에 의해 버켓(Bucket)에 피어(Peer)정보를 보관하고 이더리움의 상태정보를 상호 교환한다. 상태 정보를 비교하는 과정에서 TD(Total Difficulty)와 현재 블록 Hash를 전송한다. 블록(Block)의 동기화는 Block Headers, Block Bodies, Receipts의 과정을 순서로 패킷(Packet)을 송수신하며 수행한다[12].

Ⅲ. 노드 탐색 방법 및 실험 방법

3.1 노드 탐색 방법

노드 탐색을 위해서 우리는 탐지 노드를 이더리움 네트워크에 설치하고 이 노드로부터 데이터를 수집한다. 그림 2에서 왼쪽에 있는 "고 이더리움 클라이언트(Go Etherium Client)가 탐지 노드이다. 노드를 탐색하고 영향력 있는 노드를 찾기 위해 2단계 과정을 수행한다. 첫째, 탐지 노드의 버켓에 많은 노드 정보를 보관하기 위해서 피어 연결 최대 숫자를 기본 값보다 증가시킨다. 이더리움 클라이언트인 탐지 노드의 실행 명령어의 파라미터인 maxpeer의 값(default : 50)을 1000으로 증가시킨다. 둘째, 그림1에서 나타난 바와 같이 RLPx의 노드 찾기(Node Find) 과정에서 수집된 피어(Peer)의 이웃피어(Peer Neighbor) 데이터를 저장한다. 즉 탐지 노드가 피어 노드에게 보낸 FindNode 메시지의 응답으로 받은 이웃피어에 대한 데이터를 저장한다. 이 데이터는 탐지 노드에 연결된 피어가 알고 있는 피어의 이웃노드에 대한 데이터이며 MongoDB에 저장되고 이 데이터는 영향력 있는 노드를 탐색하기 위한 분석에 활용된다. 이더리움 클라이언트의 기본적인 특성을 그대로 유지하기 위해 Findnode 함수를 제외한 나머지 함수는 수정하지 않는다. 수집되는 피어와 이웃피어의 데이터는 IP 형태이며 피어와 이웃피어의 관계 정보도 저장된다.

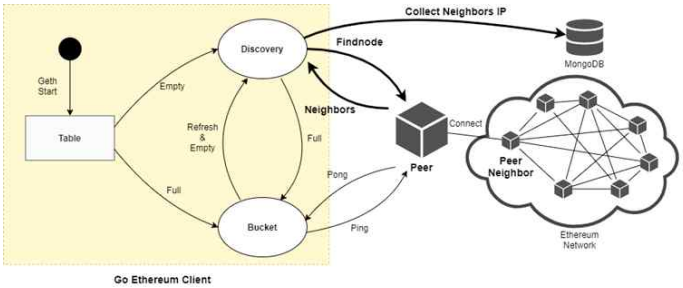


그림 2. Design of Influential Node Collection Methods

이더리움 클라이언트의 상태변화에 따른 자세한 동작 과정은 그림 2와 같다. 이더리움 클라이언트의 초기 동작 과정은 이더리움 네트워크 참여 하고 부트스트랩 노드로부터 수신 받은 노드 목록은 테이블(Table)의 버켓(Bucket)에 저장한다. 버켓에 저장된 노드 정보를 기반으로 Ping, Pong 패킷을 송수신하며 노드의 활성화 여부를 판단한다. 버켓은 피어목록이 비워져있거나 Ping메시지에 대한 응답받지 못해 피어의 목록이 갱신되는 경우 새로운 피어 찾기 기능을 수행하고 Find node 메시지 요청을 통해 피어의 이웃피어에 대한 정보를 수신 받는다[14]. 본 논문에서는 탐지 노드가 피어에 요청하는 Find node와 Neighbors 메시지를 모니터링하며 피어와 이웃피어의 매핑정보를 수집한다. 이 과정을 되도록 자주 수행해서 잠재적으로 피어가 될 가능성이 있는 정보를 많이 수집하고 피어와 이웃피어의 관계를 기반으로 노드 영향력을 분석한다[13, 15, 16].

3.2 실험 구성

실험에서 탐지 노드가 설치된 컴퓨터는 CPU가 Intel Core i7-6700 CPU@3.40GHZ * 8 CPU이고 RAM이 16GB로써 중형급 서버이며 Linux 18.04버전의 운영체제를 사용한다. 이더리움 네트워크에 참여할 때 풀 노드의 체인 데이터(Chain Data)크기는 실험당시 약 300GB정도이고 이더리움 클라이언트(Etheruem Client)의 로그저장 명령어에 따라 1TB이상의 저장 공간이 사용되었다. 원활한 동기화를 위해 1TB의 SSD의 환경을 가진 중형급 서버를 사용하여 실험하였다. 이더리움 네트워크의 참여를 위한 이더리움 Client는 대중적으로 많이 사용되는 Go-ethereum Client를 사용하였다. Client의 일부 수정을 위해 Go-ethereum은 최신버전인 1.9.12 버전을 사용하고 Go-Language는 1.13.6을 사용하였다. 탐지 노드는 파이어월과 같은 보안설정이 없는 일반 가입자 네트워크에 연결되었으며 접속 속도는 약 13Mbps이다. 실험은 2020년 3월 2일 19시부터 2020년 3월 2일 19시까지 24시간 동안 이더리움 클라이언트의 초기 노드에서 노

드 탐색 과정을 수행하였다.

Ⅳ. 노드 탐색 결과 및 영향력 분석

본 실험을 실시하여 피어와 이웃 피어의 매핑 데이터는 총 306,054개 수집되었다(FindNode 메시지의 응답으로 수집된 총 개수). 중복된 매핑을 제외한 결과 피어는 8,223개, 이들 피어의 이웃피어는 22,152개이다. 피어로 탐색된 8,223개는 실험을 실시한 하루 동안 탐색 노드와 한번이라도 연결된 노드이며 실제로 이더리움 네트워크에 연결되어 실행되고 있는 노드의 숫자이다. 우리 실험에서 찾은 이 피어들이 실제 이더리움 네트워크에 연결된 모든 노드를 탐색하였는지 확인하기 위해서 다른 이더리움 모니터링 결과와 비교했다. 기존 다른 이더리움 정보 제공 웹은 대표적으로 etherscan.io[17]와 ethernodes.org[18]에서 제공한다. 3월 2일 기준으로 가장 많은 노드 탐색 정보를 제공한 웹은 etherscan.io로 7,784개의 노드를 탐색하였고, 같은 날을 기준으로 우리의 탐지 노드가 약 500개 더 많은 노드를 탐색했다.

이웃피어 22,152개는 이더리움 네트워크에서 한번이라도 실행된 노드의 수이며 피어가 연결된 노드 개수를 의미한다. 탐색 노드에서 수집된 피어와 이웃피어의 관계를 통해 이더리움 네트워크에서 연결된 노드들의 위상과 많은 피어를 연결하며 영향력이 큰 노드를 분석했다.

본 실험의 결과로 수집된 피어와 이웃피어는 IP형태의 데이터로 저장 되며 Whois API를 사용하여 노드의 국가별 분포를 분석하였다. 이더리움 네트워크에 참여 하고 있는 노드는 110개의 국가에서 운영 중이고 75개의 IP는 추적이 불가능한 지역에서 운영 중인 결과를 확인했다.

그림 3은 수집된 이웃피어 22,152개의 정보를 바탕으로 국가별 노드 분포를 시각화한 그림과 상위 15개의 국가별 노드 분포이다. 지역적으로 분포된 노드를 시각화하기 위해 지도의 색의 농도를 5단계를 나누어 표현하고 짙은 색에 가까울수록 많은 노드를 가진 국가이다. 국가별 노드 분포는 미국, 중국, 독일, 프랑스 순서로 많은 노드를 운영하고 있다. 미국은 가장 많은 이더리움 노드를 보유 하고 있고 두 번째로 많은 중국보다 약 2배의 노드를 구축하고 이더리움 네트워크에 참여하고 있다. 이더리움 네트워크에서의 가장 영향력을 발휘하는 국가는 미국으로 전체 22,152개의 노드 중 약 42%를 차지하며 운영하고 있다. 또한 상위 15개의 국가가 전체 노드의 약 92%를 차지하며 이더리움 네트워크에 참여하여 영향력을 미치는 결과를 확인했다.

Country	Node 개수
United States	9386
China	5017
Germany	2247
France	735
Netherlands	493
Russia	437
South Korea	390
Canada	354
United Kingdom	283
Singapore	245
Hong Kong	188
Italy	184
Ukraine	140
Taiwan	121
Japan	109

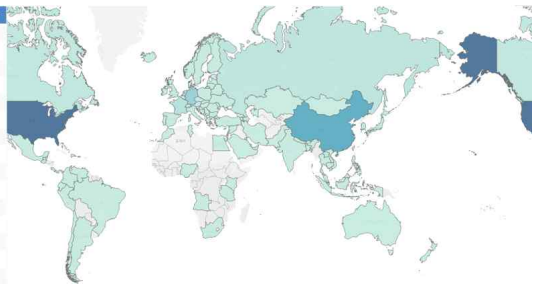


그림 3. Ethereum's Regional Influential Node Distribution and Top 15 Country List

본 실험에서 도출된 데이터는 이더리움 P2P 네트워크에 대한 연결 상태를 수집한 것은 아니다. 만약 이러한 데이터를 수집하였다면 이 데이터를 기반으로 그래프를 생성할 수 있고 이 그래프에서 기존의 영향력 분석 방법인 PageRank 등과 같은 방법으로 좀 더 정확하게 영향력을 분석 할 수 있을 것이다. 그러나 현실적으로 이더리움 P2P 네트워크의 연결 상태를 파악하는 것은 불가능하다.

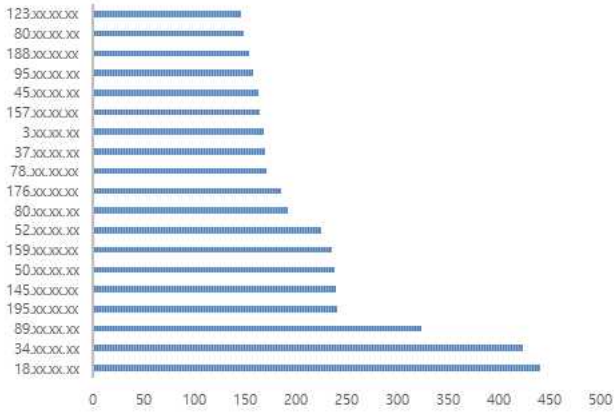


그림 4. Top 20 Influential Nodes

우리가 수집한 피어와 이웃피어의 관계에서 계산할 수 있는 노드의 영향력은 피어 관계를 맺고 있는 피어의 숫자로 단순하게 평가할 수 있다. 즉 많은 피어 관계를 맺고 있으면 이 노드는 트랜잭션이나 블록 전파에 주도적 역할을 수행하므로 많은 영향력을 가졌다고 평가할 수 있기 때문이다. 이런 영향력에 대한 관점에서 가장 많은 이웃피어를 가진 노드는 411개의 이웃피어를 가지고 있음을 확인했다. 이더리움 네트워크에 참여하고 있는 노드는 평균적으로 8개의 이웃노드를 가지고 있었다. 이더리움 클라이언트는 기본 값으로 50개의 피어와를 연결하도록 동작한다. 따라서 50개 이상의 피어를 연결하고 있는 노드는 기본 설정으로 동작하는 노드보다 영향력 관점에서 높은 영향력에 관심을 가지고 있는 노드로 생각되며 실험의 결과로 139개의 노드가 50개 이상의 이웃피어를 가지고 있었다.

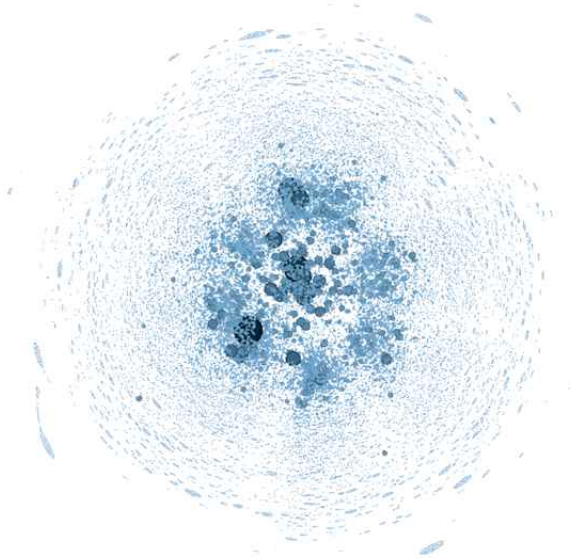


그림 5. Influence Node Snapshot

그림 4는 22,152개의 노드 IP 중 상위 20개의 영향력 있는 노드를 그래프로 나타낸 것이다(IP 주소는 정보보호를 위해서 첫째 8비트만 표시). 18.xx.xx.xx의 IP는 411개의 노드와 연결 되어 있고 가장 많은 노드와 연결되어 있다. 또한 18.xx.xx.xx, 34.xx.xx.xx와 같이 서로 다른

AS(Autonomous System)에서 이더리움 클라이언트는 운영 중이고 이더리움 네트워크에 영향력을 주고 있다.

그림 5는 하루 동안 수집된 피어와 이웃피어 정보를 통해 서로 매핑된 정보를 기반으로 영향도를 나타낸 그림이다. 이 그림은 수집 된 306,054개의 데이터를 바탕으로 이더리움 네트워크에서 활발하게 운영 중이며 영향력 있는 노드의 분포를 표시한다. 그림에서 노드의 영향력에 따라 점의 크기와 농도를 달리 표현하였다. 크기는 0~500의 단계로 나누었고 영향력이 클수록 점의 크기가 크며 점 색의 농도도 영향력이 크면 짙은 색으로 표현하였다. 점의 위치는 수집된 306,054개의 데이터를 사용하여 중앙으로 갈수록 피어와 이웃피어의 많은 연결이 되어 있는 것을 의미한다. 그림 5에서 가운데 모여 있는 크기가 크고 색의 농도가 짙은 색으로 표현된 점은 가장자리 점에 비해 개수가 작다. 이는 몇 개의 영향력 있는 노드들이 이더리움 네트워크에서 많은 참여자와 연결되어 있다는 사실을 의미한다.

Ⅴ. 결론

본 논문에서는 이더리움 네트워크에 참여하고 있는 노드를 찾기 위한 방법과 노드의 영향력 분석 결과를 제시하였다. 비트코인 네트워크와 달리 이더리움 네트워크에 대한 토폴로지 분석에 대한 연구는 아직 부족하다. 우리는 이더리움 네트워크에서 탐지 노드를 최대한 많은 노드와 연결할 수 있게 동작시키셔서 많은 정보를 이 탐지 노드로부터 수집할 수 있게 하였다. 또한 자주 반복적으로 기존 노드와의 연결을 제거하고 새로운 이웃노드와 연결을 설정하게 하여 가능한 새로운 노드를 많이 탐지하도록 동작시켰다. 이와 같은 방법을 피어와 이 피어의 이웃노드 정보를 수집하여 매핑시키고 이 데이터를 기반으로 영향력을 분석하였다. 이더리움 네트워크에서 탐지된 노드의 수는 22,152이며 이는 기존 다른 이더리움 정보 제공 웹에서 제공된 노드 숫자는 7,784개로 약 3배의 차이가 있다. 노드 영향력은 피어와 이웃피어가 관계를 맺고 있는 피어의 숫자로 단순하게 평가했다. 이 노드들은 트랜잭션이나 블록 전파에 주도적 역할을 수행하므로 많은 영향력을 준다는 사실을 의미한다. 실험을 통해 피어가 이웃피어와 50개 이상 연결되어 있는 노드 139개의 영향력이 큰 노드를 확인했다. 또한 피어와 이웃피어의 관계를 분석하여 국가별 노드 분포현황과 일일 노드 스냅샷을 통해 시각화했다.

향후 연구로는 다른 노드와 많이 연결 되어 있는 영향력 있는 노드를 중심으로 악의적인 행위를 탐지 하고 나아가 영향력 있는 노드가 이더리움 네트워크에 미치는 영향을 파악할 수 있다. 또한 노드 탐색을 통해 확장된 이더리움 네트워크의 위상(Topology)를 탐색하고 특정 블록이 전파되는 경로를 추적 할 수 있는 방법을 도출 할 수 있다.

ACKNOWLEDGMENT

이 논문은 2020년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2018R1D1A1B07050380). 또한 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539, 블록체인의 트랜잭션 모니터링 및 분석 기술개발)

참 고 문 헌

[1] Nakamoto, Satoshi. Bitcoin: A peer-to-peer electronic cash system. Manubot, 2019

[2] Buterin, Vitalik. "Ethereum: A next-generation smart contract and decentralized application platform." URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper, 2014.

[3] Ethereum RLPx, "https://github.com/ethereum/devp2p/blob/master/rlpx.md", 3월 접속

[4] 김민호, 김수진, 최훈 "블록체인의 이중지불 탐지 알고리즘", 정보과학회논문지, 제45권 제8호, 848-855, 2018 8월

[5] Lewenberg, Y., Sompolinsky, Y., & Zohar, A. Inclusive block chain protocols. In International Conference on Financial Cryptography and Data Security (pp. 528-547). Springer, Berlin, Heidelberg, 2015, January

[6] Saad, Muhammad, et al. "Exploring the attack surface of blockchain: A systematic overview." arXiv preprint arXiv:1904.03487, 2019

[7] Wang, Xu, et al. "Attack and defence of ethereum remote apis." 2018 IEEE Globecom Workshops (GC Wkshps), IEEE, 2018.

[8] Johnson, Benjamin, et al. "Game-theoretic analysis of DDoS attacks against Bitcoin mining pools." International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, 2014

[9] Miller, Andrew, et al. "Discovering bitcoin's public topology and influential nodes." et al, 2015

[10] Go-Etheruem, "https://github.com/ethereum/go-ethereum/wiki/geth" 3월 접속

[11] Maymounkov P., Mazières D. (2002) Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In: Druschel P., Kaashoek F., Rowstron A. (eds) Peer-to-Peer Systems. IPTPS 2002. Lecture Notes in Computer Science, vol 2429. Springer, Berlin, Heidelberg

[12] 명세인, 관수진, 최서윤, 이종혁, "이더리움 RLPx, Wire 프로토콜 분석", 한국통신학회 학술대회논문집, 269-270, 2018

[13] Y Gao, J Shi, X Wang, Q Tan, C Zhao, "Topology Measurement and Analysis on Ethereum P2P Network", IEEE Symposium, 2019

[14] 명세인, 이종혁 "이더리움 노드 탐색 프로토콜 분석", 한국통신학회 논문지, 제43권 12호, 2018

[15] Kim, Seoung Kyun, et al. "Measuring ethereum network peers." Proceedings of the Internet Measurement Conference 2018, 2018

[16] Essaid, Meryam, Sejin Park, and Hongteak Ju. "Visualising Bitcoin's Dynamic P2P Network Topoogy and Performance." 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2019.

[17] Ethereum Monitoring Site, "https://etherscan.io/nodetracker", 3월 2일 접속

[18] Ethereum Monitoring Site, "https://www.ethernodes.org/", 3월 2일 접속

지분 증명 합의 알고리즘의 발전에 관한 연구

최원석, 우종수, 홍원기
포항공과대학교

{ws4583, woojs, jwkhong}@postech.ac.kr

A Study on Proof-of-Stake Consensus Algorithms

Wonseok Choi[1], Jongsoo Woo[2], James Won-Ki Hong[1]
[1]Department of Computer Science and Engineering, POSTECH
[2]Graduate School of Information Technology, POSTECH

요 약

작업 증명 합의 알고리즘(Proof of Work)은 채굴이라는 혁신적인 방안을 통해 비잔틴 장군 문제와 이중 지불 문제에 대한 해결책을 제시했다. 하지만 점점 경쟁이 치열해지면서 작업 증명 방식의 채굴 난이도는 이에 따라 증가했고, 연산에 높은 수준의 장비와 과도한 양의 전력이 소모되며 큰 에너지 낭비가 발생했다. 지분 증명 합의 알고리즘(Proof of Stake)은 이런 작업 증명 합의 알고리즘의 막대한 에너지 소모에 대한 대안으로써 등장했다. 지분 증명은 검증자들이 보유하고 있는 암호화폐의 지분에 따라 의사결정 권한을 주는 합의 알고리즘들을 통틀어 지칭하는 단어이다. 초기의 지분 증명 합의 알고리즘은 작업 증명 합의 알고리즘과 융합된 형태였으나 순수하게 지분 증명 방식만 사용하는 합의 알고리즘들도 등장했고, 어느 합의 알고리즘이 가장 안전하고 효율적인가에 대한 연구는 계속해서 진행중이다. 최근 이더리움의 창시자인 비탈릭 부테린은 이더리움 2.0 에 대한 계획을 발표했고, 그 핵심 내용 중 하나는 이더리움의 작업 증명에서 지분 증명으로의 전환이었다. 그리고 구체적인 지분 증명의 구현 방식에 대한 내용은 지속적으로 업데이트 되고 있다. 본 논문에서는 초기에 출현한 지분 증명 합의 알고리즘들과 현재 많은 연구가 진행중인 이더리움 2.0 에서의 지분 증명에 대한 분석을 통해 지분 증명 방식이 발전되어 온 과정과 각각의 장단점을 파악하여 앞으로의 지분 증명의 방향성에 대하여 제시하고자 한다.

Ⅰ. 서 론

비트코인 [1]은 2008 년 익명의 개발자 혹은 개발자 그룹인 사토시 나카모토에 의해 처음 공개되었다. 비트코인은 작업 증명(PoW, Proof of Work) 합의 알고리즘을 통하여 비잔틴 장군 문제[2]와 이중 지불 문제[3]를 해결하려 한 최초의 암호화폐이자 가장 유명하고 성공적인 암호화폐이다. 작업 증명 합의 알고리즘에서는 채굴(Mining)이라는 과정을 통해 암호화폐의 거래들이 담긴 블록을 생성한다. 채굴 과정에서 모든 채굴자(Miner)들은 지정된 채굴 난이도에 따라 목표 값보다 작은 해시 값을 생성하는 논스(Nonce) 값을 반복해서 찾는다. 가장 먼저 위 조건에 해당하는 논스 값을 찾은 채굴자는 블록을 생성할 권한을 가지게 되고 블록을 생성한 대가로 일정량의 암호화폐를 얻게 된다.

이러한 작업 증명 방식은 51% 공격[4]을 효과적으로 방지할 수 있기에 긍정적으로 평가받았으나 시간이 지나며 작업 증명 합의 알고리즘에 대한 문제점들도 대두되었다. 가장 큰 문제점은 작업 증명의 채굴 과정에서 오는 막대한 전력 소모였다. 이는 에너지 낭비와 더불어 채굴자들의 단합으로 거대한 채굴 풀(Mining pool)[5]들이 생겨나며 블록 체인의 초기 목표에서 벗어나 오히려 네트워크가 중앙화 되는 현상을 가져왔다.

작업 증명 합의 알고리즘의 이런 문제점을 해결하기 위해 지분 증명(PoS, Proof of Stake)[6] 합의 알고리즘이 등장했다. 지분 증명 합의 알고리즘은 검증자들이 보유한 암호화폐의 지분에 비례하여 의사결정 권한을 가지게 되는 합의 알고리즘들을 통틀어 지칭하는 단어이다. 지분 증명 방식은 작업 증명과 달리 계산 집약적인 작업을 필요로 하지 않기 때문에 작업 증명에서의 막대한 전력 소모 문제를 해결하였고, 작업 증명에 비해 51% 공격을 더욱 효과적으로 막을 수 있다. 하지만, 지분 증명도 여전히 여러 안전성 및 확장성 등의 문제점이 제기되고 있고 이에 대한 연구가 지속적으로 진행중이다.

본 논문에서는 이러한 지분 증명 합의 알고리즘의 출현과 현재 가장 이슈가 되고 있는 이더리움 2.0 [7] 에서의 지분 증명 합의 알고리즘에 대해 분석함으로써 지금까지의 지분 증명의 발전에 대해 정리하고 앞으로 지분 증명 합의 알고리즘이 나아가야할 방향성에 대하여 제시하고자 한다.

Ⅱ. 지분 증명 합의 알고리즘

1. 피어 코인(Peercoin)

피어 코인[6]은 지분 증명 합의 알고리즘을 도입한 최초의 암호화폐로 평가받고 있다. 피어 코인에서는 코인 나이를 도입한 지분 증명 방식과 작업 증명을 융합하여 순수 작업 증명에서의 막대한 에너지 소모 문제를 해결했다. 피어 코인의 지분 증명에서는 피어 코인을 30 일 이상 소유하면 블록 생성에 참여할 수 있게 되고 90 일 이상 소유할 경우 블록 생성에 참여할 확률이 최대가 된다. 코인 나이는 코인을 소유한 기간과 소유한 코인의 수의 곱으로 계산되며 코인 나이가 클수록 블록 생성에 필요한 해시 값을 찾기가 쉬워진다. 작업 증명에서는 블록 생성에 대한 보상이 별도로 존재하지만 지분 증명에서는 거래 수수료외에는 보상이 존재하지 않는다는 것이 큰 차이점이다. 분기가 발생하면 사용된 코인 나이가 큰 체인이 메인 체인으로 선택되며 블록 생성에 성공했다면 일정 비율의 코인을 보상으로 받게 되며 코인 소유 기간이 초기화된다. 피어 코인에서는 이러한 방식의 지분 증명과 작업 증명 방식을 함께 사용하여 에너지 소모를 줄이고 유저들의 참여를 격려한다.

한편, 지분 증명에서는 전체 코인의 51% 이상을 보유해야 51% 공격이 가능하기 때문에 작업 증명에 비해 51% 공격이 더욱 어렵다는 장점 역시 있다. 하지만 피어 코인에서는 코인 보유 기간을 악의적으로 늘려 새로운 체인을 메인 체인으로 만드는 장거리 공격이 가능한 문제점이 있는데 이를 막고자 피어 코인에서는 일정 블록마다 체크포인트를 두어 체크포인트 이전에서는 포크가 발생하지 않도록 했는데 체크포인트는 개발자가 임의로 선정하기 때문에 이것이 피어 코인의 탈중앙화성을 떨어뜨리게 되었다.

2. Nxt

Nxt[8]는 순수하게 지분 증명 방식으로만 구현된 최초의 암호화폐로 평가받고 있다. 피어 코인에서는 지분 증명 방식과 작업 증명 방식을 함께 사용하는 것과 달리 Nxt 에서는 지분 증명 합의 알고리즘만 사용한다. 블록 생성 과정에 참여하기 위해서는 한 계정에 1,440 개의 블록이 생성되는 동안 지분을 걸어 두어야 한다. 이 조건을 만족한 계정은 활성 계정이 되며 블록 생성을 위한 기본 목표 값보다 작은 해시 값을 찾아 내면 블록을 생성할 수 있게 된다. 기본 목표 값은 이전 블록의 목표 값과 생성 시간에 따라 정해진다. 이때, 어느 계정이 블록을 생성할 권리를 갖고, 블록 충돌이 발생했을 때 어느 블록을 유효한 블록으로 처리할 것인가는 이전 블록 생성 후 경과 시간과 계정의 잔액에 영향을 받는다. 즉, 이전 블록 생성 후 오랜 시간이 지날수록, 계정에 많은 코인을 보유할수록 블록을 생성하기 쉬워진다. 모든 계정의 보유 코인은 공개된 정보이기 때문에 이는 다음 블록의 생성자를 예측하기 쉽다는 의미이기도 하다. Nxt 에서는 블록 생성에 대한 대가로 블록 내부의 거래들의 수수료를 가질 수 있다.

하지만, 이 방식에도 문제가 발생할 수 있다. 이는 기본 목표 값이 정해지는 과정에서 발생한다. 기본 목표 값은 이전 블록의 값을 바탕으로 생성되기 때문에 이전 블록의 생성자가 의도적으로 자신에게 유리한 값을 삽입하게 된다면 독점이 발생한다는 것이다. 이는 다른 여러 지분 증명 합의 알고리즘에서도 발생할 수 있는 문제로 이 문제의 가장 핵심이 되는 점은 블록 생성을 위한 랜덤 값의 신뢰성이다.

	피어 코인	Nxt
합의 알고리즘	PoW + PoS	PoS
블록 생성 기준	코인 나이	계정 잔액
블록 생성 참여 조건	코인 나이 30 일 경과	지난 1,440 블록 생성 기간 동안 1000NXT 이상 소유

표 1. 피어 코인 Nxt 비교

3. 이더리움 2.0 (Ethereum 2.0)

비탈릭 부테린은 이더리움 2.0 을 발표하며 작업 증명에서 지분 증명으로의 합의 알고리즘의 전환을 예고했다. 앞서 설명했듯, 이 역시 작업 증명 방식의 막대한 에너지 소모와 중앙화 된 채굴자, 또한 51% 공격 등의 문제를 해결하기 위함이었다. 하지만 지분 증명 합의 알고리즘에도 앞서 소개한 안전성 문제나 랜덤성 문제 외에 다른 문제가 존재한다. 지분 증명은 작업 증명과 같이 증명 과정에 필요한 에너지 소모가 없기 때문에 검증자 입장에서는 모든 체인에 투표하는 것이 이득이고 이로 인해 진정한 합의에 이를 수 없다는 Nothing at stake 문제이다. 이더리움 2.0 에서는 이러한 문제점을 잘못된 체인에 투표했을 때 처벌하는 방식을 이용하여 해결 하고자 한다.

이더리움 2.0 의 발전 과정에서 구체적인 합의 알고리즘에 대해 많은 논의가 있었다. 그 과정에서 작업 증명과 지분 증명이 혼합된 형태로 비탈릭 부테린이 주도하는 Casper FFG(friendly finality gadget)[9]와 순수 지분 증명을 사용하고 블라드 잠피르가 주도하는 Casper CBC(correct by construction)[10]로 나뉘어 졌다.

Casper CBC 는 수학적 증명과 같은 탄탄한 기초부터 시작해서 궁극적으로는 필요한 기능이 모두 구현되고 오류가 없도록 하는 correct by construction 기법을 도입한 합의 알고리즘이다. Casper CBC 는 지분 증명에 관한 연구로부터 시작되었지만 점차 그 영역을 넓혀 일반적인 합의 알고리즘에 대한 연구로 확장되었고 그 중 블록체인에서의 지분 증명을 집중적으로 다루는 연구는 Casper TFG(the friendly GHOST)[11]에서 진행되었다. Casper TFG 에서의 핵심 내용은 지분 증명에서 블록 분기가 발생했을 때의

처리 방법이다. 일반적인 방법으로는 가장 긴 체인, 혹은 가장 많은 지분이 걸린 체인이 메인 체인이 되지만 Casper TFG 에서는 LMD GHOST(latest message driven greedy heaviest observed subtree) 프로토콜을 사용한다. 이 프로토콜에서는 검증자들의 가장 최근의 메시지를 바탕으로 가장 많은 검증자들이 선호하는 체인을 메인 체인으로 삼는다. 이는 가장 긴 체인을 메인 체인으로 삼는 방식과 유사해보이나 잦은 분기가 발생할 때 소수의 검증자가 악의적인 행동을 하더라도 다수의 검증자를 절대 이길 수 없다는 점에서 큰 장점을 가진다.

Casper FFG 는 현재 이더리움의 작업 증명에서 지분 증명으로 넘어가기 위한 단계적인 과정이기에 작업 증명 기반 위에 지분 증명이 추가된 형태로 향후 작업 증명에 해당하는 부분을 바꿔 나가는 계획으로 진행되었다. Casper FFG 는 일반적인 작업 증명 과정에서 일정 블록 수 마다 검사 지점이 존재하고 이 지점에서 지분 증명 기반의 검증자들에 의한 완결성 검증이 존재한다. 일정 블록 수 마다 검증자들은 자신의 지분을 이용해 블록 분기에 투표를 하고, 전체 지분의 2/3 이상이 투표한 블록이 확정되게 된다. 자신이 투표한 블록이 확정되면 그에 따른 보상을 받게 되고, 이중 투표와 같은 악의적인 행동이 발각되면 지분을 몰수당하는 처벌을 받게 된다.

	Casper TFG	FFG
합의 알고리즘	PoS	PoW + PoS
분기 선택 규칙	LMD GHOST	가장 높은 검사 지점을 보유한 체인

표 2. Casper TFG FFG 비교

Gasper[12]는 Casper FFG 와 Casper TFG 에서 사용된 LMD GHOST 프로토콜을 합친 형태로 순수 지분 증명 합의 알고리즘이며 이더리움 2.0 에서 적용할 가장 이상적인 방안으로 최근 새롭게 제안되었다. Gasper 에는 시간을 슬롯으로 구분하며 슬롯 마다 블록이 채워져 있을 수 있고 하나의 epoch 는 여러 개의 슬롯으로 구성되어 있다. Epoch 의 경계는 Casper FFG 에서의 검사 지점으로 사용되며 epoch 마다 검증자들은 슬롯 수만큼의 위원회로 랜덤하게 배정이 되고, 각 슬롯 마다 위원회 중 한 명의 검증자가 블록을 제안하며 나머지 검증인 들은 투표를 한다. 그리고 블록 분기가 발생하면 LMD GHOST 를 약간 변형한 형태인 Hybrid LMD GHOST 프로토콜에 따라 가장 많은 투표를 받은 블록을 선택하게 된다. Gasper 에서는 Casper FFG 와 유사한 방식으로 검증자가 악의적인 행동을 하지 않고 올바른 블록을 생성하는데 성공하면 보상을 받게 되고, 악의적인 행동을 보였다면 지분을 몰수당하는 처벌을 받게 된다.

Ⅲ. 결 론

본 논문에서는 작업 증명 합의 알고리즘의 대안으로 등장한 지분 증명 합의 알고리즘의 발전 과정과 그 응용사례들에 대하여 소개하였다. 지분 증명은 과도한 전력소모라는 작업 증명의 문제점을 해결하기 위해 등장했고, 51% 공격에 대한 문제를 더욱 효과적으로 해결하였다. 하지만, 여전히 51% 공격에 대한 완전한 해결책을 가져다 주진 못하였고, 작업 증명과는 또다른 중앙화성과 불완전한 랜덤성, Nothing at stake 문제 등 많은 새로운 문제가 나타나 이에 관한 연구들이 진행중이다. 새롭게 등장하는 지분 증명 합의 알고리즘들은 작업 증명 등 다른 합의 알고리즘과 융합된 형태들도 여럿 존재하며 저마다 앞의 문제들을 해결하기 위한 독자적인 해결책을 제시하고 있다. 이더리움 2.0 역시 현재 이더리움이 사용하고 있는 작업 증명의 문제점을 해결하기 위해서 합의 알고리즘을 지분 증명으로 전환하려는 계획을 가지고 있다. 이 과정에서 단계적으로 작업 증명과 지분 증명을 함께 사용하지만, 궁극적으로는 지분 증명으로 완전히 대체하고자 한다. 또한, Casper 합의 알고리즘과 GHOST 프로토콜을 이용하여 공평성과 안전성을 더욱 확보하고자 하였으며, Nothing at stake 문제를 위한 처벌 제도 역시 도입할 계획이다.

본 논문에서 소개한 지분 증명 합의 알고리즘들 외에도 세상에는 다양한 종류의 지분 증명 합의 알고리즘들이 존재한다. 그리고 각 알고리즘들은 신뢰성, 보안성, 실용성 등 저마다 각각의 목적을 달성하고자 하며 실질적으로 이러한 모든 특성들을 완벽히 달성하기는 어렵기에 어떤 알고리즘이 더 우월하다고 판별하기 어렵다. 이처럼 앞으로의 지분 증명 합의 알고리즘 역시 각자의 독자적인 목적에 맞게 51% 공격에 대한 좀 더 높은 안전성, 투명하고 편중되지 않은 랜덤한 값의 생성 및 Nothing at stake 문제 등에 대해 기존의 알고리즘보다 더욱 좋은 해결책을 가지고 등장할 것이다. 하지만, 궁극적으로는 이러한 알고리즘들을 다양한 방면에서 활용할 수 있도록 모든 알고리즘들을 아우를 수 있는, 혹은 필요에 따라 전환이 가능한 알고리즘이 필요해질 것이다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 임(No.2018-0-00539)

참 고 문 헌

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." *URL: https://bitcoin.org/bitcoin.pdf (accessed: 15.04. 2020)* (2008).

[2] Lamport, Leslie, Robert Shostak, and Marshall Pease. "The Byzantine Generals

Problem." *Concurrency: the Works of Leslie Lamport*. 2019. 203-226.

[3] Chohan, Usman W. "The double spending problem and cryptocurrencies." *Available at SSRN 3090174* (2017).

[4] Ye, Congcong, et al. "Analysis of security in blockchain: Case study in 51%-attack detecting." *2018 5th International Conference on Dependable Systems and Their Applications (DSA)*. IEEE, 2018.

[5] Eyal, Ittay, and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable." *International conference on financial cryptography and data security*. Springer, Berlin, Heidelberg, 2014.

[6] King, Sunny, and Scott Nadal. "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake." *URL: https://decred.org/research/king2012.pdf (accessed: 15.04. 2020)* (2012).

[7] Buterin, Vitalik. "Ethereum 2.0 mauve paper." *Ethereum Developer Conference*. Vol. 2. *URL: https://cdn.hackaday.io/files/10879465447136/Mauve%20Paper%20Vitalik.pdf (accessed: 15.04. 2020)* (2016).

[8] Popov, Serguei. "A probabilistic analysis of the nxt forging algorithm." *Ledger* 1 (2016): 69-83.

[9] Buterin, Vitalik, and Virgil Griffith. "Casper the friendly finality gadget." *arXiv preprint arXiv:1710.09437* (2017).

[10] Zamfir, V., et al. "Introducing the minimal CBC Casper family of consensus protocols." *DRAFT v1. URL: https:// github.com/cbc-casper/cbc-casper-paper/blob/master/cbc-casper-paper-draft.pdf (accessed: 15.04. 2020)* (2018).

[11] Zamfir, Vlad. "Introducing casper the friendly ghost." *Ethereum Blog URL: https://blog. ethereum. org/2015/08/01/introducing-casperfriendly-ghost (accessed: 15.04. 2020)* (2015).

[12] Buterin, Vitalik, et al. "Combining GHOST and Casper." *arXiv preprint arXiv:2003.03052* (2020).

해시레이트와 트랜잭션 분석을 통한 나카모토 사토시 비트코인 규모 추정 연구

신혜영, 주홍택

계명대학교 컴퓨터공학과

yeoung@stu.kmu.ac.kr, juht@kmu.ac.kr

A Study on Estimation of Bitcoin Scale by Satoshi Nakamoto through Hash-Rate and Transaction Analysis

hye-yeong Shin, Hongtaek Ju

Dept. of Computer Engineering, Keimyung University

요 약

비트코인은 거래 당사자의 익명성이 보장되기 때문에, 특정 개인이 소유한 비트코인의 양을 추정할 수 없다. 하지만 불법거래나 돈세탁과 같은 합법적이지 않은 거래에 참여한 사용자가 소유한 비트코인의 양을 추정하는 것은 필요한 작업이다. 본 논문은 불법거래 당사자는 아니지만, 비트코인 시스템을 만든 나카모토 사토시가 보유한 비트코인(BTC)을 채굴 경향과 트랜잭션에 저장된 거래 정보를 분석하여 추측한 결과이다. 비트코인 초기 네트워크에는 사토시가 설치한 채굴 노드만이 존재하였고 새로운 노드가 추가되기 전까지 오직 사토시만 채굴하고 있었을 것이다. 새로운 노드의 참여는 해시레이트(Hash Rate)의 변화로 알 수 있고, 새로운 노드가 참여하기 전까지의 채굴 수수료는 모두 사토시가 소유한 것으로 추정할 수 있다. 또한, 트랜잭션을 분석하여 사토시가 소유한 지갑에서 사용된 주소들을 그룹화시키고 이 주소들이 보유한 비트코인의 양을 계산하여 사토시가 보유한 비트코인의 양을 추정하였다. 해시레이트 변화와 트랜잭션 분석을 통한 주소 그룹화로 사토시가 보유한 비트코인을 추정한 연구의 학술 가치는 결과보다도 연구 방법에 있다. 이 연구를 통해 특정인이 소유한 비트코인 양을 추정하는 방법을 발전시키면 불법거래에 사용된 거래 금액을 추정하는데 적용될 수 있기 때문이다.

I. 서론

비트코인(Bitcoin)은 2008년 10월 나카모토 사토시(Nakamoto satoshi)라는 익명의 인물이 작성한 논문[1]을 기반으로 구현된 최초의 암호화폐 시스템(Cryptocurrency system)이며, 2009년 1월 3일 사토시가 제네시스 블록(0번 블록)을 생성하였다. 비트코인 네트워크에 참여하는 모든 노드는 동일한 블록체인 데이터를 갖고 블록체인에 기록된 데이터를 조작할 수 없으므로 데이터의 투명성(Transparency)과 데이터의 불변성(Immutability) 그리고 무결성(Integrity)이 보장된다. 또한, 참여자의 개인정보와 관련한 정보가 기록되지 않기 때문에 거래 당사자의 익명성(Anonymity)이 보장된다.

비트코인 거래에 참여한 사용자의 익명성이 보장되는 특징을 이용하여 비트코인(BTC; bitcoin)은 불법거래나 돈세탁 등 합법적이지 않은 거래에 사용되곤 한다. 불법거래에 참여한 거래 당사자가 누구인지 알아내기 위해서는 당사자가 어떤 비트코인 지갑 주소를 소유하고 있는지 알아야 가능하다. 이러한 사실 정보는 비트코인에서 발생한 정보만으로 알아내는 것이 불가능하므로, 불법거래가 이루어지는 다크 웹(Dark Web) 등에서 알아낼 수 있다. 불법거래에 사용된 주소를 알아낸 이후에는 거래가 기록된 트랜잭션을 분석하여 총 거래 규모 산정이나 거래 후의 비트코인의 행방을 추적할 수 있다.

포렌식은 불법적인 행위에 대한 수사를 위해 숨겨진 사실을 찾아내고 증거 자료를 수집하는 과정으로써, 위와 같은 포렌식을 비트코인에 적용한 것을 비트코인 포렌식이라고 할 수 있다. 더 확장해서는 블록체인 포렌식이라 할 수 있으며 블록체인 포렌식 기술은 이미 불법거래 탐지나 돈세탁 규모 추정 등에 활용되고 있고 여러 회사가 이 기술을 보유하고 있음을 공표하고 있지만, 그 내부적인 기술을 공개하지 않고 있다. 대표적인 블록체인 포렌식 전문회사로는 Chainanlysis나 Elliptic이 있다[15. 16].

본 논문에서는 불법거래 당사자는 아니지만, 확실한 비트코인 소유자인 사토시가 보유한 비트코인 양을 추정함으로써 기초적인 블록체인 포렌식을 실시하고 그 방법과 결과를 제시한다. 먼저, 제네시스 블록이 생성된 후 일정 기간 동안 오직 나카모토 사토시만이 비트코인 네트워크에 참여하였을 것이라는 사실을 근거로 사토시가 소유한 비트코인의 규모를 추정한 결과를 제시한다. 비트코인에서 해시레이트(Hash Rate, 또는 해시 파워)는 채굴에 참여한 모든 채굴자(마이너)의 컴퓨팅 능력을 나타내는 지표이다. 채굴자가 많이 참여할수록 해시레이트는 커지며 해시레이트가 커지면 10분 간격으로 채굴이 되어야 하나 이 간격이 줄어들게 된다. 간격이 줄어들게 되면 채굴 난이도(Difficulty)를 높여서 10분 단위로 채굴이 되도록 조절한다. 초기 비트코인 시스템에서는 사토시만 채굴했을 것이며, 그동안의 해시레이트는 일정하게 유지되었을 것이다. 또한, 해당 기간의 채굴 수수료는 모두 사토시가 소유한 것으로 생각할 수 있다. 이와 같은 사실을 근거로 해시레이트 변화를 분석하여 사토시의 비트코인 소유 양을 추정하였다.

Dorit 등은 트랜잭션에 기록된 거래의 패턴과 주소(address) 정보를 이

용하여 같은 지갑에 있을 가능성이 있는 특정 주소들을 모아 하나의 주소 집합을 생성했다[2]. 이 집합에 포함된 주소는 한 지갑에서 생성된 주소이거나 한 사람이 소유한 것으로 추정된다. 또한, 주소들과 관련 있는 트랜잭션들을 분석하여 해당 지갑의 전체 재정활동에 대해 분석했다. 물론 이 방법은 특정 지갑에 있을 가능성이 있는 주소들을 찾을 수는 있지만, 익명성으로 인해 지갑의 주인에 대해서는 알지 못한다. 본 논문에서는 사토시 소유가 확실한 주소로부터 시작하여 Dorit 방법으로 주소를 그룹화하였으며 그 결과로 사토시가 소유한 비트코인 규모를 추정한 결과를 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 기본적인 용어에 대한 설명하고, 3장에서는 관련 연구에 대해 기술 한다. 4장에서는 해시레이트 측정 방법과 측정 결과를 설명한다. 또한, 5장에서는 트랜잭션을 분석하여 주소를 그룹화하여 사토시가 보유한 코인의 양을 추정한 방법과 결과를 제시한다. 마지막으로 6장에서는 분석 결과와 향후 연구에 관해 설명하며 본 논문을 마친다.

II. 이론적 배경

2.1. 엑스트라-논스(ExtraNonce)

채굴자가 주어진 논스값을 전부 대입해도 목표값보다 작은 블록 헤더 해시값을 찾지 못한다면, 채굴자는 블록 헤더의 timestamp(채굴 시간) 또는 트랜잭션의 머클트리(Merkle Tree)를 바꾸어 새로운 논스 값을 찾는데 이를 엑스트라-논스(ExtraNonce)라 한다.

2.2. bits, 목표값(Target value)

목표값(T)은 블록 헤더 해시값의 상한선을 의미하며, 32byte(256bit)로 표기된다. 채굴로 생성된 블록의 해시값은 항상 $[0, T)$범위 안에 있어야 한다. 목표값은 실수의 부동소수점 표시 방식으로 크기가 4byte(32bit)로 압축되어 블록 헤더에 bits라는 필드에 기록된다. bits 필드의 맨 왼쪽 1바이트는 목표 값의 길이를 비트로 나타낸 것이며 나머지 3개 비트는 목표값의 최상위 3개 바이트를 나타낸다.

예를 들면, 제네시스 블록의 bits 값 인 $0x1d00ffff$일 때 목표값은 **그림 1**과 같다.

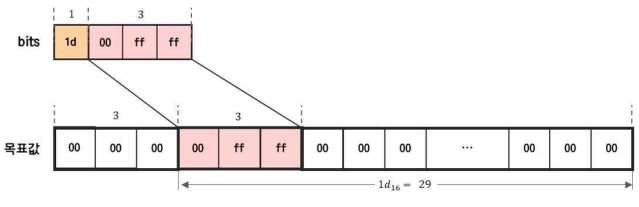


그림 1. bits 값이 $0x1d00ffff$일 때 목표값

bits값의 처음 1byte는 bits 값을 목표값으로 변환한 결과의 길이가 총 29바이트(Ox1d)라는 의미이다. 그리고 bits값의 나머지 3byte는 목표값의 처음 3byte를 의미한다. bits값을 목표값으로 변환하는 공식은 (수식 1)과 같다.

$$T = C \cdot 2^{8(E-3)}$$

(여기서 T는 목표값, C는 계수 값, E는 지수 값)

수식 1. bits값을 목표값으로 변환하는 공식

제네시스 블록의 bits 값으로 목표값을 구하는 과정은 다음과 같다.

$bits = 0x1d00ffff$

$\therefore C = ffff_{16}$ 이고 $E = 1d_{16} = 29$

$\therefore$ 목표값의 길이= 29byte

$$\begin{aligned} \therefore T &= ffff_{16} \cdot 2^{8(1d_{16}-3)} \\ &= ffff_{16} \cdot 2^{8(29-3)} \quad (\because 1d_{16} = 29_{10}) \\ &= ffff_{16} \cdot 2^{208} \end{aligned}$$

(여기서 T는 목표값, C는 계수 값, E는 지수 값)

따라서, 목표값의 범위는 $\left(0, ffff_{16} \cdot 2^{208}\right]$ 이고 해당 범위에서 구할 수 있는 해시값의 개수는 2^{224} 개다.

목표값은 약 10분마다 블록이 생성되도록 조절되며, 2016개 블록이 생성된 시간을 기준으로 변경된다. 블록 생성시간이 2016개 블록을 생성하는 동안 평균적으로 10분보다 작으면 목표값을 작게 하여 해시값을 찾기 어렵게 하고 10분보다 크면 목표값을 크게 하여 해시값을 찾기 쉽게 한다. 이처럼 변경된 목표값은 bits로 변환되어 블록에 저장된다.

bits값과 목표값이 변경될 때 난이도 값도 변한다. 난이도는 하나의 블록을 생성하는 것이 제네시스 블록과 비교했을 때, 얼마나 어려워졌는지에 대한 수치이다. 난이도가 커질수록 목표값의 범위는 작아진다. 난이도는 아래 공식(수식 2)으로 구할 수 있다. 이때, $T_C \neq 0$ 이다.

$$D = \frac{T_G}{T_C}$$

(여기서 T_G는 제네시스 블록의 목표값, T_C는 현재 블록의 목표값)

수식 2. 블록 난이도 공식

현재 블록의 bits 값이 $0x1d00ffff$일 때 난이도는 1이며, 비트코인 블록 채굴 난이도의 최솟값이 1임을 알 수 있다[6].

즉, 2016 블록 생성 주기마다 평균 블록 생성시간을 계산하고 이 결과로 다음 목표값과 bits값을 변경한다. 또한, 이것이 난이도 변화로 이어진다. 이상적으로 약 10분씩 2016개의 블록을 생성하기 위해선 2주 정도의 시간이 소요된다.

2.3. 해시레이트(Hash rate)

해시레이트는 비트코인 블록을 채굴한 컴퓨터의 초당 연산 속도이다. 컴퓨터의 연산 속도가 빠를수록, 즉 해시레이트가 높을수록 더 많은 블록을 채굴할 수 있다.

블록 해시값은 32byte로 표현되는데, 32byte(256bit)에서는 총 2^{256}개의 값을 구할 수 있다. 또한, 난이도가 D 일 때 목푯값의 범위는 $\left(0, T_G/D\right]$이므로 ($\because$난이도는 $T_C \neq 0$ 일 때, $D = T_G/T_C$ 이므로 $T_C = T_G/D$), 해당 범위에서 구할 수 있는 해시값의 개수는 $2^{224}/D$ 개다. 따라서 난이도가 D 일 때 블록 해시값을 구할 확률은 $\frac{2^{224}}{D} \div 2^{256} = \frac{2^{224}}{D \times 2^{256}} = \frac{1}{D \times 2^{32}}$이다[3]. 즉, 해시값 표현 범위 내에서 조건에 맞는 블록 헤더의 해시값을 구할 수 있는 확률은 $1/D \times 2^{32}$이다.

난이도가 D 일 때 작업증명이 성공할 때까지 평균적인 시행횟수는 기하분포를 이용하여 구할 수 있다. 작업증명의 결과는 '성공'과 '실패'로 나눌 수 있기 때문에, 작업증명과정을 기하확률 변수로 생각할 수 있다. 따라서, 난이도가 D 일 때 블록 해시값을 구할 확률은 $1/D \times 2^{32}$이므로 한 블록을 채굴할 때까지 평균적으로$(D \times 2^{32})^{-1} = D \times 2^{32}$ 만큼의 작업증명 과정을 거쳐야 조건에 맞는 해시값을 구할 수 있다.

만약, 난이도가 1이고 하나의 블록을 생성할 때 10분이 걸렸다면 그때의 해시레이트는 $\frac{1 \times 2^{32}}{600} \approx 7Mhashs/\text{sec}$ 이다[7].

Ⅲ. 관련 연구

Har Finney(Harold Thomas Finney Ⅱ)는 사토시 외에 처음으로 비트코인 네트워크에 참여한 비트코인 초기 사용자 겸 채굴자이다[8]. 그는 70여개의 블록을 채굴하였으며, 사토시와 비트코인 초기 소프트웨어 문제와 관련된 메일[9. 10]을 주고받으며 문제를 해결했다. 주고받은 메일 대부분의 내용은 Har Finney가 비트코인 소프트웨어의 버그를 발견하면 사토시가 버그를 수정하는 것이었다. 또한 Har Finney는 사토시에게 10BTC를 받았는데[8. 10], 이것은 비트코인 최초의 거래이다. 거래 내용이 기록된 트랜잭션은 170번 블록에 포함되어있다. 이때 사토시가 Har Finney에게 비트코인(BTC)을 전송할 때 사용한 주소는 9번 채굴 주소이다.

Har Finney와 사토시가 주고받은 메일[9]에 따르면 비트코인 초기 소프트웨어 버전(v.0.1.0)은 소프트웨어 실행문제와 비트코인 네트워크 소켓 통신 문제 등이 있었음을 알 수 있다. 네트워크 소켓 통신의 문제를 해결한 버전(v.0.1.3)을 사토시는 1월 12일(UTC 기준) 저녁에 공개했다. 하지만, 또 다른 문제로 인해 해당 버전이 정상적으로 작동하는 것은 1월 13일부터이다(1월 3일 이후 10일경과).

2013년 4월, Sergio Demean Lerner는 제네시스 블록이 생성되고 14일(2009년 1월 3일부터 1월 16일) 동안 사토시가 유일한 마이너였다면, 2010년 1월 25일까지 사토시가 유일한 채굴자일 것이라고 주장했다. 해당 기간의 해시레이트가 7*Mhash*/sec 보다 낮게 유지되기 때문에 다른 참여자가 없을 것으로 추측하며 사토시는 최소 1,148,800BTC를 소유했을 것이라고 주장했다[11].

Sergio는 자신의 주장을 뒷받침하는 기술적인 문서를 웹에 공지하였다[12]. 해당 문서에서는 사토시만이 유일한 채굴자였을 것이라고 주장한 증거를 엑스트라 넌스 값을 중심으로 설명하고 있다. 2009년에 생성된 블록의 엑스트라 넌스가 일정한 패턴으로 증가하기 때문에 한 명의 채굴자가 22,976개 이상의 블록(이때, 채굴 수수료는 1,148,800BTC)을 채굴했을 것으로 추측하는 것이다.

BITMEX RESEARCH에서는 Sergio가 주장한 실험을 재연 결과를 발표 하였다[13]. BITMEX는 2009년에 한 명의 채굴자가 많은 블록을 채굴했을 수 있다는 Sergio의 주장에는 동의했다. 하지만, 엑스트라 넌스 값이 일정한 패턴으로 증가하는 것은 채굴 노드가 동일한 비트코인 소프트웨어를 실행하거나 비슷한 사양의 하드웨어를 사용했을 경우 동일한 패턴을 생성할 수도 있다는 점에서 논리적인 오류가 있다고 지적했다.

A. Pinar Ozisik등은 블록 채굴 노드들이 작업증명 상황을 주변 노드에 지속적으로 전파하고, 그것으로 해시레이트를 더욱 정밀하게 측정하는 방법에 대해서 제시했다[3]. 분석 결과, A. Pinar Ozisik등이 새롭게 제시한 방법이 기존 블록체인에 저장되어있는 데이터로 측정한 해시레이트보다 채굴 노드들의 변화에 대해 자세히 알 수 있다. 결론적으로, 해시레이트로 채굴 노드들의 변화를 더 정확히 알 수 있는 방법을 제시하였다.

Dorit Ron and Adi Shamir는 트랜잭션과 주소(address) 정보를 이용하여 같은 지갑에 있을 가능성이 있는 특정 주소들을 모아 하나의 주소 집합을 생성했다[2]. 또한, 주소들과 관련 있는 트랜잭션들을 분석하여 해당 지갑의 전체 재정활동에 대해 분석을 했다. 분석 결과로 가장 많이 비트코인을 갖고 있던 주소가 무엇인지 알 수 있었다. 또한, 트랜잭션과 주소와의 관계를 시각적으로 표현하여 거래 흐름을 쉽게 알 수 있도록 분석 결과를 도출했다. 이 연구에서 사용한 방법을 적용하여 사토시가 보유한 코인의 양을 추정하였다.

Ⅳ. 해시레이트를 통한 연구 방법

이번 장에서는 블록 생성 시 소요된 시간을 이용하여 해시레이트를 구하고, 채굴 노드의 변화에 대해 분석한다.

4.1. 연구 방법

본 실험에서는 비트코인 초기 네트워크 해시레이트 변화율과 해시레이트가 증가한 시점을 확인하기 위해 제네시스 블록부터 94038번 블록(UTC 기준 2010-11-26 23:58:55 생성)까지의 데이터를 수집했다. 2010년 11월 27일 최초의 채굴 풀(Mining pool)인 slush pool[15] 정식으로 운영되었기 때문에 11월 26일까지 생성된 블록 데이터를 수집했다.

따라서 본 실험에서는 제네시스 블록부터 94038번 블록 데이터의 생성시간인 타임스탬프(timestamp)를 이용하여 해시레이트를 계산한다. 난이도가 D 일 때 한 블록을 채굴할 때까지 평균적으로 $D \times 2^{32}$ 만큼의 작업증명 과정을 거쳐야 조건에 맞는 해시값을 구할 수 있다. 따라서, 난이도가 D 일 때 블록 하나를 생성하는데 n분이 걸렸다면 $n \times 60$초로 변환하여 초당 해시레이트를 계산한다. 그때의 해시레이트는 **수식 3**과 같다.

$$\frac{D \times 2^{32}}{n \times 60}\ (\text{sec})$$

수식 3. 블록 하나의 해시레이트

즉 블록 생성시간으로부터 해시레이트를 계산할 수 있다. 블록 생성시간은 블록 헤더에 기록되며 블록 생성시간이 비록 주변 채굴 노드와의 시간 차이를 고려한 네트워크 시간이지만 비트코인 초기에는 채굴 노드가 많지 않으므로 시간 변동이 적고 블록 간 상대적인 시간이 정확할 것이라고 가정한다.

수식 3을 이용하여 100번 블록의 해시레이트를 구하는 과정은 다음과 같다. 우선, 100번 블록을 생성하는 데 어느 정도의 시간이 소요되었는지 알기 위해, 99번 블록과 100번 블록의 타임스탬프를 확인한다.

$$timestamp_{99} = 1231660146$$

$$timestamp_{100} = 1231660825$$

$$\therefore n = timestamp_{100} - timestamp_{99}$$
$$= 1231660825 - 1231660146$$
$$= 679\ (\text{sec})$$

$$\therefore hashrate = \frac{D \times 2^{32}}{n} = \frac{1 \times 2^{32}}{679} = 6325430.48 = 6.32\ Mhash/\text{sec}$$

따라서, 100번 블록을 생성하기 위해 소요된 해시레이트는 6.32 *Mhash*/sec 이다.

4.2. 연구 결과 및 분석

해당 구간(제네시스 블록부터 94038번 블록)에서는 총 31번 난이도가 변경되었다. 첫 번째로 난이도가 변경된 것은 32256번 블록(UTC 기준 2009-12-30 06:11:04 생성)이다. 처음으로 난이도가 변경되기 전에 총 16번의 난이도 변경 주기가 발생했었지만, **그림 2**를 보면 알 수 있듯 난이도가 변경될 만큼의 충분한 해시레이트가 발생하지 않았다. **그림 2**에서의 expeted hashrate는 해당 난이도에서의 이상적인 해시레이트이고, hashrate는 실제 2016개의 구간에서 발생한 평균 해시레이트이다.

또한, 첫 번째로 난이도가 변경된 이후에는 **그림 3**과 같이 난이도가 지속해서 증가한 것을 확인할 수 있다. 이것은 비트코인 네트워크에 참여한 채굴자의 수가 점차 증가한 것으로도 생각할 수 있다.

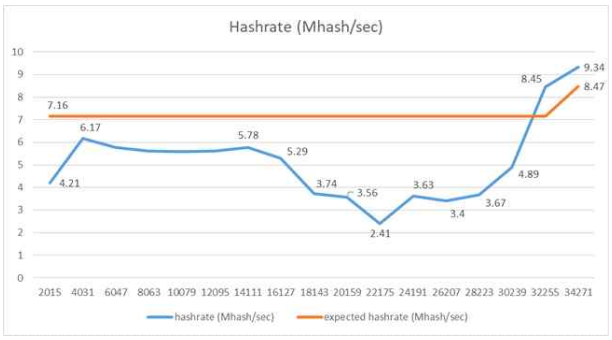


그림 2. 첫 번째로 난이도가 변경되기 전 난이도 변경 주기 때의 해시레이트

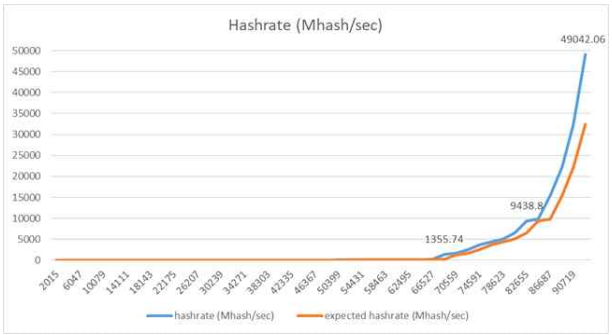


그림 3. 채인 풀이 운영되기 전까지 난이도 변경 주기 때의 해시레이트

해시레이트가 증가한 시점을 분석하기 위해 32256번 블록 이전의 해시레이트를 분석하던 중, 2297번 블록(UTC 기준 2009-01-30 1:13:03 생성) 이후 생성된 2298번 블록(UTC 기준 2009-01-30 1:12:17 생성) 생성시간이 2297번 블록보다 46초 빠른 것을 발견했다. 이것은 블록을 생성한 채굴 노드의 시스템 설정값이 달라서 발생하는 현상인데, 이는 곧 새로운 노드가 추가된 것임을 알 수 있다[16].

따라서, 제네시스 블록부터 2297번 블록까지는 시스템 설정값이 동일한 채굴 노드가 작동했을 것이며 사토시가 홀로 채굴했을 것으로 생각할 수 있다. **그림 4**는 제네시스 블록부터 2297번 블록까지의 블록 당 해시레이트다.

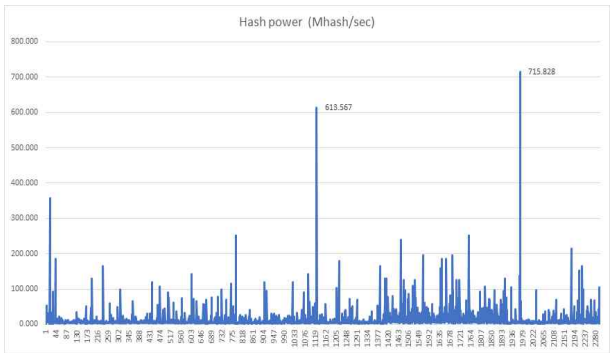


그림 4. 제네시스 블록부터 2297번 블록까지의 블록 당 해시레이트

작업증명의 결과는 '성공'과 '실패'로 나눌 수 있으므로, 작업증명 과정을 기하확률로 생각할 수 있다. 따라서, 블록 당 해시레이트를 분석하는 것은 확률적인 결과이므로 분석하기 힘들며 어떠한 특징도 찾지 못했다. 하지만, 하루당 평균을 구하는 경우 하루 동안의 해시레이트의 변화를 확인 가능하므로 비트코인 네트워크에 변화를 확인할 수 있다. 하루 당 평균은 **그림 5**과 같다.

그림 5는 2009년 1월 하루 동안의 평균 해시레이트 그래프이다. 실제로 네트워크가 정상적으로 작동한 1월 13일에는 2.87 *Mhash*/sec였던 해시레이트가 1월 14일 6.4 *Mhash*/sec로 약 2배가량 증가한 것을 알 수 있다.

그림 5. 2009년 1월 하루 평균 해시레이트

해시레이트가 2배 증가한 1월 14일은 비트코인 네트워크에 Har Finney가 채굴자로 참여한 시점이다. 해시레이트가 증가하기 전인 1월 13일까지 생성된 385번 블록까지 사토시가 홀로 채굴했음을 알 수 있다. 따라서 총 19,300BTC(386X50BTC)를 채굴 수수료로 지금 받았음을 알 수 있다.

비트코인 네트워크의 해시레이트가 2배 증가한 것은 채굴 노드의 해시파워가 2배 증가한 것이고 사토시의 채굴 기회가 반으로 줄어든 것으로 추정할 수 있다. 새로운 채굴 노드가 누구의 것인지(사토시의 것일 수도 있음) 모르지만 이후부터는 사토시의 채굴 수수료 수입은 반을 줄어들었을 것이다. 이후, 하루 평균 해시레이트가 1월 13일 해시레이트의 n배라면, 사토시의 채굴 확률은 1/n으로 추측할 수 있을 것이다.

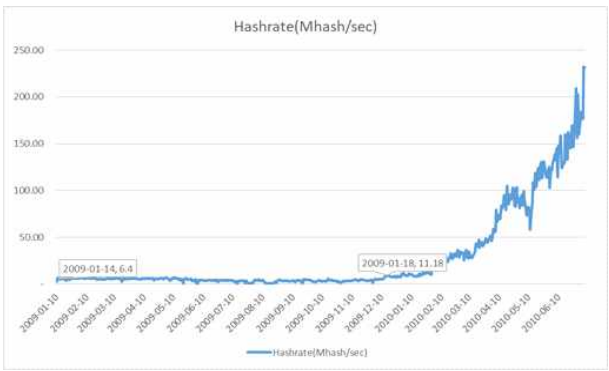


그림 6. 하루 별 평균 해시레이트

그림 6은 난이도가 처음으로 증가한 이후, 해시레이트는 지속적으로 증가했다. 2009년 12월 18일의 해시레이트는 11.18 *Mhash*/sec로 2009년 1월 14일 해시레이트(6.4 *Mhash*/sec)보다 약 1.8배 높으며 1월 13일 해시레이트(2.87 *Mhash*/sec)보다는 약 3.9배 높다. 1월 13일 이후로 채굴 노드 해시파워가 4배 가량 증가한 것을 알 수 있다. 사토시의 채굴 확률은 약 1/4로 생각할 수 있다. 이와 같은 방식으로 하루 별 평균이 1월 13일의 평균 해시레이트의 n배이면 1/n의 확률로 채굴을 했을 것이라고 가정했고, 이때 사토시는 총 94,039개의 블록 중 23,572개의 블록을 채굴 했을 것으로 추측할 수 있다.

따라서 사토시가 총 23,572개의 블록을 사토시가 채굴했다면. 1,187,600BTC(23,572X50BTC)를 채굴 수수료로 지금 받았음을 알 수 있다. 따라서, 이후에도 사토시가 채굴에 참여했다고 가정할 경우, 비트코인 네트워크에 참여하는 채굴 노드 개수가 늘어나면 늘어날수록 채굴할 수 있는 확률은 반비례 할 것이다.

따라서, 해시레이트를 통해 사토시는 최소 1,187,600BTC를 갖는 것으로 추정할 수 있다. 이후에도 사토시가 채굴에 참여했다고 가정한다면 1,187,600BTC 이상의 비트코인을 소유하고 있음을 알 수 있다.

V. 트랜잭션을 통한 연구 방법

이번 장에서는 트랜잭션과 주소를 이용하여 같은 지갑에 있을 가능성이 있는 주소들을 찾고 주소들의 비트코인(BTC) 잔액 정보를 수집한다.

5.1. 연구 방법

제네시스 블록부터 623,379블록(UTC 기준 2020년 3월 28일까지 생성된 데이터)의 블록 데이터를 비트코인 getblock RPC 메서드[14]를 사용하여 수집하였다. 수집한 블록 데이터는 **표 1**과 같다.

Name	Type	Description
block height	numeric	블록 높이
block hash	string	블록 해시값
nTx	numeric	트랜잭션 개수
tx	array of string	트랜잭션 id 리스트
bits	string	블록의 bits 값
difficulty	numeric	블록 생성 난이도
nonce	numeric	블록 넌스 값
timestamp	numeric	블록 생성 시간(UTC 표현)
date time	string	블록 생성 날짜
previous block hash	string	이전 블록 해시
next block hash	string	다음 블록 해시

표 1. 비트코인 블록 데이터구조

트랜잭션 데이터의 상세 정보를 더욱 쉽게 찾기 위해 수집된 블록 데이터에서 블록 데이터의 tx의 txid와 getrawtransaction RPC 메서드를 이용해 각 트랜잭션의 상세 정보를 수집·저장하였다.
트랜잭션의 상세 정보를 이용해 트랜잭션 거래에서 사용된 지갑 주소를 주소 데이터에 저장했다. 주소 데이터구조는 **표 2**와 같다. 주소 데이터에서는 각 주소의 현재 비트코인(BTC) 잔액 정보를 확인할 수 있다. 또한, 주소 데이터의 'vin tx list' 또는 'vout tx list' 항목에서 주소가 참여한 다른 트랜잭션들을 확인할 수 있다. 본 연구에서는 하나의 트랜잭션에서 입력 주소로 사용한 모든 지갑 주소를 같은 지갑에서 생성된 주소로 추정하므로 'vin tx list'를 참조한다.

Name	Type	Description
address	string	지갑 주소
BTC balnace	numeric	주소에 보관된 코인
send value	numeric	주소가 송금한 비트코인(BTC) 총합
receive value	numeric	주소가 받은 비트코인(BTC) 총합
vin nTx	numeric	'vin tx list' 개수
vout nTx	numeric	'vout tx list' 개수
vin tx list	array of string	비트코인을 송금한 내용이 기록된 트랜잭션 id 리스트
vout tx list	array of string	비트코인을 받은 내용이 기록된 트랜잭션 id 리스트
first send time	string	처음 비트코인을 송금한 내용이 기록된 블록과 블록 생성시간
last send time	string	마지막으로 비트코인을 송금한 내용이 기록된 블록과 블록 생성시간
first receive time	string	처음 비트코인을 받은 내용이 기록된 블록과 블록 생성시간
last receive time	string	마지막으로 비트코인을 받은 내용이 기록된 블록과 블록 생성시간

표 2. 주소 데이터

트랜잭션 데이터구조는 **표 3**과 같다. 트랜잭션 데이터에서는 트랜잭션에 입·출력 주소로 사용된 주소들을 확인할 수 있다. 트랜잭션의 입력 주소는 'vin address list' 항목에서 확인할 수 있다. 입력 주소가 비트코인을 송금한 내용이 기록된 트랜잭션(vin 트랜잭션)은 'vin tx list' 항목에서 확인할 수 있다.

Name	Type	Description
txid	string	트랜잭션 id
block height	numeric	트랜잭션이 포함되어있는 블록 해시값
vin address list	array of string	트랜잭션 입력 주소 리스트
vout address list	array of string	트랜잭션 출력 주소 리스트

표 3. 트랜잭션 데이터구조

본 실험에서는 **그림 7**과 같은 방법으로 같은 지갑에 속한 다른 주소를 추적한다. **그림 7**은 **주소 A**와 같은 지갑에 속한 다른 지갑 주소를 찾는 과정 중 하나이다. **먼저, 주소 A**의 주소 데이터 'vin tx list' 항목을 참

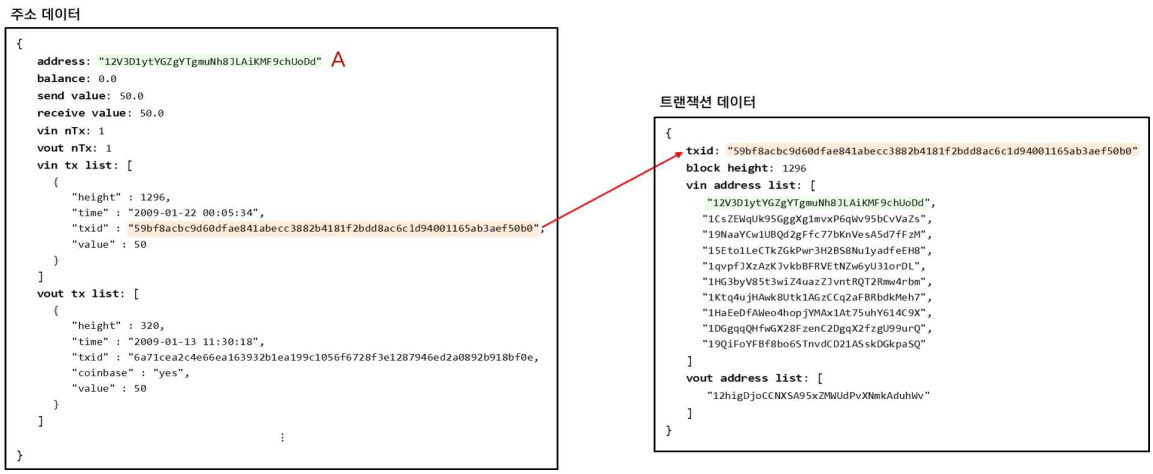


그림 7. 특정 주소와 같은 지갑에 속한 다른 지갑 주소를 찾는 과정

조한다(vin 트랜잭션 리스트를 확인하기 위함). 주소 데이터의 'vin tx list'에서 트랜잭션 데이터(txid)를 찾은 후, 해당 트랜잭션의 트랜잭션 데이터에서 'vin address list' 항목을 확인한다. 트랜잭션 데이터의 'vin address list'에서 **주소 A** 외에 다른 주소들이 기재되어있다면, 그것이 **주소 A**와 같은 지갑에 속한 지갑 주소이다. **그림 7**의 트랜잭션 데이터에서는 **주소 A**와 같은 지갑에 속한 지갑 주소는 9개임을 알 수 있다.

Har Finney는 사토시에게 10BTC를 받은 것은 비트코인 최초의 거래였고, 해당 거래 내용이 기록된 트랜잭션[1]은 170번 블록에 포함되어있다[8. 10]. 이때 사토시가 Har Finney에게 비트코인(BTC)을 전송할 때 사용한 주소는 9번 채굴 주소[2]이다. 따라서, 사토시의 주소임을 알 수 있는 것은 제네시스 블록과 9번 블록의 채굴 주소이다. 제네시스 블록의 채굴 주소에서는 어떠한 vin 트랜잭션도 발생하지 않았다. 하지만, 9번 블록의 채굴 주소는 서로 다른 5개의 주소로 비트코인을 전송했다(총 5번의 vin 트랜잭션 발생).

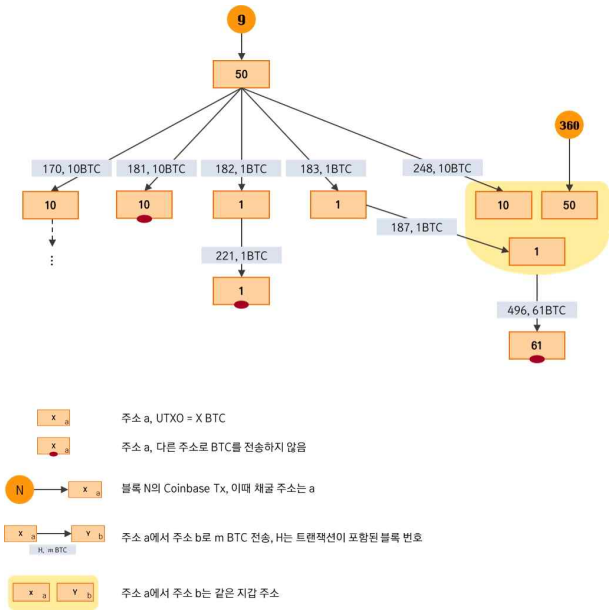


그림 8. 9번 채굴 주소로 비트코인을 받은 서로 다른 4개의 주소 트랜잭션 그래프

그림 8은 Har Finney의 주소를 제외한 서로 다른 4개의 주소 트랜잭션 거래를 그래프로 도식화한 것이다. **그림 8**에서 9번 블록 채굴 주소로부터 비트코인(BTC)을 전송받은 서로 다른 4개 주소 중 3개의 주소에서는 최소 한 건의 트랜잭션이 발생했다. 하지만 비트코인(BTC)을 송신 받은 주소에서는 어떠한 트랜잭션도 발생하지 않았기 때문에, 4개의 주소를 이용해서 같은 지갑에 있는 주소를 찾기 힘들다.

그림 9는 9번 블록 채굴 주소로부터 비트코인(BTC)을 전송받은 Har Finney의 주소[3]트랜잭션 그래프이다. **그림 9**의 **트랜잭션 A**의 입력 주소 정보는 **표 4**와 같다.

index	mining block	mining address
1	419	19CkFSEiHB5UVah3fvZD17qk48m82TfAp8
2	528	12oRSUW6UVYNCUk8yyrFvbpbJw7uia31sA
3	651	14nELEgJL95NU3BKi5D734ysFuVUCEjLWg
4	685	1ADTuxdhYePCW9PEvkCKnib6jmbg6mKXdz
5	752	1DdhxvYwVrnUP4xcWaquUnmZ177HVeUT2L
6	803	1DCcZbNtttndh6tvoK8xRTB5BUMdfL51aL′
7	819	17LDnEt8ggVjH43QdjhZ4FhkXC9zXgk48b
8	842	18QQUbHJhyzFDVKhEHPhHFZF3qdBmp74eB
9	869	1CbDYwNDp5aA5FVmCf2Z95kjkMCgoBr2X8
10	966	1AoWi2xSyQoWuxrYrdr1PPVi5PxA8caNGC
11	1042	1HQkgckTBytqaGmwyQgqTU5aptuZ7qFiyp
12	1063	1NdLg6FNKyFXyRfmFssW4DSgaWRzscYXUg
13	1153	1BSgXqBzNHFfdqRHBPzLD23CFXpzhqpRrb
14	1195	1GSAXEQEW7st7prSuz89x6DjKEcLUFSDCJ
15	1233	1AHJbmcNpaxrgwDc12XFXnUSfpED5ksSiH
16	1316	13evwJtdkEp2wcmv5xtqE7mF6pr5bFrbxw
17	1376	14DFY3RsdrJ5U6CCgatYssM2gKAiyBm4mE
18	1471	1Jy5dQmet2xg1Pk5VYZE2hSWu8StK6aQWZ
19	1525	1BoBh9JKnHVj1JAy2GS4LeA3yHyN6ra2qE
20	1614	1B2EqNLFj7hoSWJupZLQrGMd2unu8doTjc

표 4. 그림4의 트랜잭션 A의 입력 주소 정보

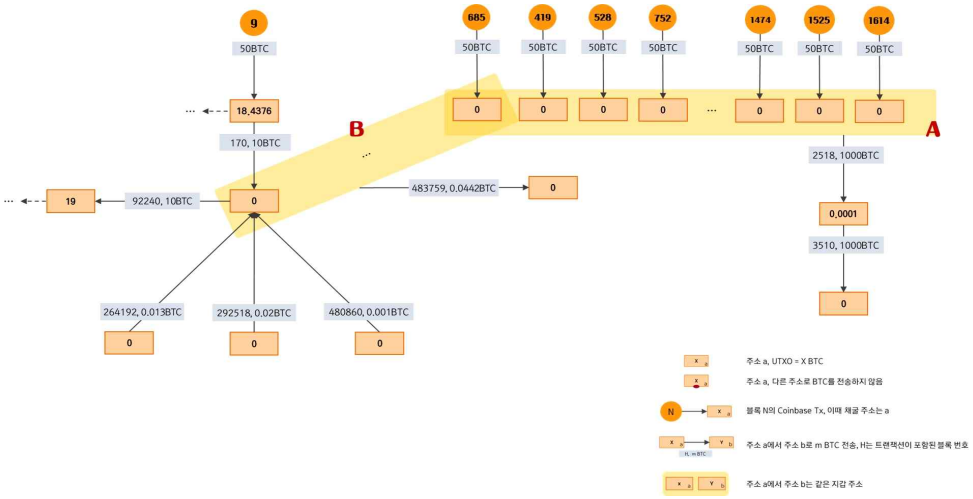


그림 9 Har Finney의 주소 트랜잭션 거래 그래프

1) **txid**: f4184fc596403b9d638783cf57adfe4c75c605f6356fbc91338530e9831e9e16
2) 12cbQLTFMXRnSzktFkuoG3eHoMeFtpTu3S
3) 1Q2TWHE3GMdB6BZKafqwxXtWAWgFt5Jvm3

index	mining block	mining address	Used block	Input txid List	
1	9	12cbQLTFMXRnSzktFkuoG3eHoMeFtpTu3S	170	1	f4184fc596403b9d638783cf57adfe4c75c605f6356fbc91338530e9831e9e16
			181	2	a16f3ce4dd5deb92d98ef5cf8afeaf0775ebca408f708b2146c4fb42b41e14be
			182	3	591e91f809d716912ca1d4a9295e70c3e78bab077683f79350f101da64588073
			183	4	12b5633bad1f9c167d523ad1aa1947b2732a865bf5414eab2f9e5ae5d5c191ba
			248	5	828ef3b079f9c23829c56fe86e85b4a69d9e06e5b54ea597eef5fb3ffef509fe
2	78	1AiBYt8XbsdyPAELFpcSwRpu45eb2bArMf	130667	6	5c85ed63469aa9971b5d01063dbb8bcdafd412b2f51a3d24abf2e310c028bbf8
3	235	1PmhUgoVLtTgdcvfi1cwwSTvPvFC67bCQs	94012	7	e8160a014fbff8386548f40205d540ef92ce8207ff4ac0446d6e591c6cf28f2c
4	268	1DNdPgBZRWjDj1JbVZQEYMv7jvqJF7R4Py	586	8	4d6edbeb62735d45ff1565385a8b0045f066055c9425e21540ea7a8060f08bf2
5	286	1Jhk2DHosaaZx1E4CbnTGcKM7FC88YHYv9	524	9	d71fd2f64c0b34465b7518d240c00e83f6a5b10138a7079d1252858fe7e6b577
6	309	1627A2DbCtVVykWVJmdQz2ERwkw4uiEL22	131723	10	a1be8f4fd67f305bbdd03be77104dda20680b934f12acb66c239a944bd5cb798
7	317	1E5npMTmh9PnBHxCacuqzAZRGEybkxPMAq	586	11	6bf363548b08aa8761e278be802a2d84b8e40daefe8150f9af7dd7b65a0de49f
8	320	12V3D1ytYGZgYTgmuNh8JLAiKMF9chUoDd	1296	12	59bf8acbc9d60dfae841abecc3882b4181f2bdd8ac6c1d94001165ab3aef50b0
9	329	153h6eE6xRhXuN3pE53gWVfXacAtfyBF8g	593	13	e36f06a8dfe44c3d64be2d3fe56c77f91f6a39da4a5ffc086ecb5db9664e8583
			1056	14	90ff15e5a80593977fb2f6666de2860584d39ebc3a41f65a0a1fdc3a851aefda
10	357	18KrJNtPVu6LWRNPQReqF29iFm7vDhirMk	728	15	6f7cf9580f1c2dfb3c4d5d043cdbb128c640e3f20161245aa7372e9666168516
11	360	18SH9vwx24L5cTabfkgtGMjF8A56pD9AUJ	496	16	a3b0e9e7cddbbe78270fa4182a7675ff00b92872d8df7d14265a2b1e379a9d33
12	361	12wej8tANWruTQFEhWFJtTkjNLt76pE268	130667	17	5c85ed63469aa9971b5d01063dbb8bcdafd412b2f51a3d24abf2e310c028bbf8
13	372	1PQjPXAtSZfUiiQeD4nafQ7HKx7J1kYQ4J	130673	18	5644b0002fe4c18afc769826d7655f4972ec45dbd33e0afcbf75624591d345cb
14	394	1JV9ZQX6cCd4YrUtHHQm9iDL6cMxP3oQ3J	1184	19	c445fc181b4570fa107006f31c68132df2df84160f6dc783a21e10f4497e3589
15	407	1PEsXxy7kVSPL3Sqw9q4HPkbM368tGTYcx	1056	20	90ff15e5a80593977fb2f6666de2860584d39ebc3a41f65a0a1fdc3a851aefda
16	413	15ATbhqgqxkGen8ZLsb2BEbQzw3ymdzbQ9	130673	21	5644b0002fe4c18afc769826d7655f4972ec45dbd33e0afcbf75624591d345cb
17	417	1ELmSkQWnqgbBZNzxAZHts3MEYCngqRBeD	586	22	4d6edbeb62735d45ff1565385a8b0045f066055c9425e21540ea7a8060f08bf2

표 5. 제네시스 블록부터 418번 블록 사이에 있는 채굴 주소 중
거래가 발생한 주소 정보

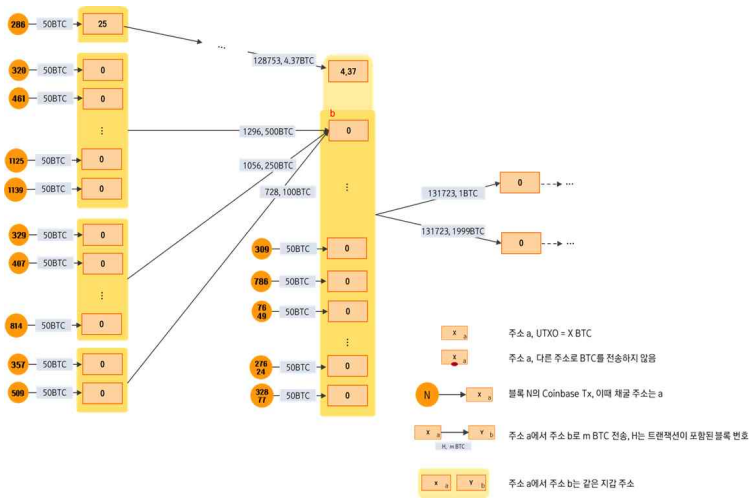


그림 10. 주소 b와 여러 채굴 주소들과의 관계

트랜잭션 A의 입력 주소 중 685번 블록은 **트랜잭션 B**에서 Har Finney의 주소와 함께 입력 주소로 사용되므로, **표 4**에 기재되어있는 채굴 주소는 Har Finney의 것이다. 또한, **그림 9**와 **표 4**를 통해 알 수 있는 것은 Har Finney가 채굴한 여러 개의 블록 중 제일 먼저 채굴한 것은 419번 블록(UTC 기준 2009-01-14 06:22:48 생성)이다. 이는 비트코인 버전(v.0.1.3)이 정상적으로 작동하기 시작한 시기와 일치한다.

이것을 통해 사토시가 제네시스 블록부터 418번 블록까지 채굴한 것을 알 수 있다. 따라서, 본 실험에서는 제네시스 블록부터 418번 블록 채굴 주소와 트랜잭션을 이용하여 사토시가 소유한 지갑 주소와 비트코인의 규모를 추정한다.

5.2. 연구 결과 및 분석

표 5는 제네시스 블록부터 418번 블록(UTC 기준 2009-01-14 06:20:25 생성)까지 조사하여, 총 419개의 채굴 주소 중 비트코인(BTC)을 다른 주소로 전송한 주소들을 나타낸 것이다. 총 17개의 채굴 주소에서 채굴 수수료로 받은 코인을 다른 주소로 전송하였고, 해당 거래 내용이 기록되어있는 트랜잭션은 **표 5**의 Used block에 포함되어있다.

표 5의 채굴 주소를 이용하여 이들 주소와 동일한 지갑에서 생성된 것으로 추정되는 주소를 찾는다. 우선 동일한 입력에 사용한 지갑 주소를 찾기 위해 **표 5**의 채굴 주소의 주소 데이터 'vin tx list' 항목에서 각 채굴 주소가 입력으로 사용된 모든 트랜잭션을 찾는다. **표 5**의 총 17개의 채굴 주소에서 발생한 vin 트랜잭션은 총 22개이며 **표 5**의 'Input txid List'에서 확인할 수 있다.

일부 채굴 주소는 개별적으로 트랜잭션을 발생시키기도 했지만, 나머지 다른 채굴 주소는 같은 트랜잭션에서 입력 주소로 사용되기도 했다. 채굴 주소가 같은 트랜잭션에서 입력 주소로 사용된 경우를 제외하면 총 19개의 트랜잭션이 발생하였음을 알 수 있다.

표 5에 기재된 총 19개의 트랜잭션(중복 제외)의 입력부에 기록되어있는 입력 주소들은 총 129개(중복 제외, **표 5**의 채굴 주소 제외)이다. 이들 129개의 주소는 **표 5**의 17개 주소와 같이 트랜잭션에서 사용되었으므로 사토시가 소유한 지갑에서 생성된 주소로 그룹화시킬 수 있다. 사토시는 제네시스 블록부터 418번 블록까지 사토시가 채굴하였으므로 419개 채굴 주소와 129개의 주소 총 531개의 주소를 갖고 있음을 알 수 있다.

다시 이들 129개의 주소 데이터 'vin tx list' 항목에서 각 채굴 주소가 입력 주소로 사용된 모든 트랜잭션을 찾고, 해당 트랜잭션의 입력 주소들을 찾는다. 129개의 주소에서 발생한 트랜잭션은 총 39개이며, 트랜잭션의 입력부에 기록되어있는 주소들은 총 429개(중복 제외)이다.

이처럼 새로 발견한 지갑 주소들이 참여한 트랜잭션을 찾고, 해당 트랜잭션의 입력 주소로 참여한 주소를 찾는 과정을 반복한다. 새로운 트랜잭션 또는 주소를 찾을 때까지 이와 같은 과정을 반복함으로써 사토시의 지갑 주소를 추측한 결과 그의 주소는 총 920개이며 이들 주소에 보관된 코인 중에서 아직 사용되지 않은 코인(UTXO: Unspent Transaction Output)의 총합은 61,004.27BTC이다.

또한, **그림 10**은 사토시의 지갑 **주소 b**[4]의 트랜잭션 그래프이다. 지갑 **주소 b**는 2018년 4월 5일까지 거래를 했다는 점에서 2년 전까지 사토시가 계속 활동하고 있음을 알 수 있다. **주소 b** 또는 다른 주소에서 만약 추가적인 거래가 발생한다면 사토시가 보유한 주소와 비트코인(BTC)양은 변화할 수 있기 때문에, 추가적인 거래가 발생하는지 예의 주시할 필요가 있다.

VI. 결론 및 향후 계획

본 논문에서는 해시레이트와 트랜잭션 분석을 통해 나카모토 사토시의 비트코인 규모 추정 연구를 시행하였다. 사토시는 불법거래 당사자는 아니지만, 확실한 비트코인 소유자로써 그가 소유한 비트코인 양을 추정함으로써 블록체인 포렌식의 기초적인 방법과 결과를 제시하였다.

해시레이트를 통한 연구 방법에서는 채굴 풀이 정식으로 운영되기 전인 2010년 11월 26일까지 생성된 블록 데이터를 수집하여 비트코인 초기 네트워크의 해시레이트를 측정하였다. 그 결과 새로운 채굴자가 추가함으로써 해시레이트가 증가하고, 해시레이트가 일정하게 유지되다가 2009년 후반부에 증가하는 것을 확인할 수 있었다. 따라서, 해시레이트를 통한 연구 방법을 통해 사토시가 최소 762,900BTC을 소유하고 있음을 추정할 수 있다. 채굴 풀이 운영됨에 따라 해시레이트를 통한 연구 방법으로 채굴자가 소유하고 있는 코인을 추정하는 것은 현재 힘들 것으로 생각되어진다. 하지만 이 연구 방법을 코인 추정이 아닌 이중 지불, 51% 공격 등의 다양한 연구와 접목시켜 발전시킬 수 있을 것으로 생각된다.

4) 12higDjoCCNXSA95xZMWUdPvXNmkAduhWv

마지막으로 본 논문의 트랜잭션을 통한 연구방법에서는 제네시스 블록부터 623,379블록(UTC 기준 2020년 3월 28일까지 생성된 데이터)의 블록 데이터까지 수집한 후 분석하였다. 해당 방법을 통해 하나의 주소가 참여한 여러 트랜잭션 거래 정보로 같은 지갑에서 관리되는 주소를 찾을 수 있었고, 그 결과 사토시의 지갑 주소 920개와 해당 지갑 주소가 61,004.27BTC을 보유하고 있음을 알 수 있다. 또한, 사토시의 주소로 추정되는 **그림 10**의 **지갑 주소 b**는 2018년 4월 5일까지 거래를 했다는 점에서 사토시가 계속 활동하고 있음을 알 수 있다. 추가적으로 **지갑 주소 b**에서 다른 거래가 발생한다면 사토시의 지갑 주소와 보유하고 있는 비트코인의 양이 변화할 것으로 예상된다. 다음 연구에서는 트랜잭션을 통한 연구 방법을 발전시켜 불법적인 행위에 사용된 여러 주소들과 그 금액을 추정하고자 한다.

ACKNOWLEDGMENT

이 논문은 2020년도 정부(교육부)의 재원으로 한국연구재단의 지원과 (NRF-2018R1D1A1B07050380) 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539, 블록체인의 트랜잭션 모니터링 및 분석 기술개발)

참 고 문 헌

[1] Stoshi Nakamoto, "Bitcoin: A peer-to-peer electronic cash system", 2008

[2] Dorit Ron and Adi Shamir, "Quantitative analysis of the full bitcoin transaction graph," in Financial Cryptography and Data Security - 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers, 2013, pp. 6 - 24.

[3] A. Pinar Ozisik, George Bissias, Brian Levine, "Estimation of Miner Hash Rates and Consensus on Blockchains," arXiv preprint arXiv:1707.00082, 2017. Accessed:2017-09-25.

[4] Chainalysis, Available Online, "https://www.chainalysis.com/"

[5] Elliptic, Available Online, "https://www.elliptic.co/"

[6] Xiaojing Zhang, Rui Qin, Yong Yuan and Fei-Yue Wang, "An Analysis of Blockchain-based Bitcoin Mining Difficulty: Techniques and Principles", 2018 Chinese Automation Congress (CAC), 2018

[7] Hash rate, Available Online, "https://en.bitcoin.it/wiki/Difficulty#What_network_hash_rate_results_in_a_given_difficulty.3F"

[8] Bitcoin and me(Hal Finney), Available Online, "https://bitcointalk.org/index.php?topic=155054.msg1643833#msg1643833"

[9] (2019). Kicking the Hornet's Nest: The Complete Writings, Emails, and Forum Posts of Satoshi Nakamoto, the Founder of Bitcoin and Cryptocurrency, Mill Hill Books, 39p - 61p

[10] Cryptography mail, Available Online, "https://www.mail-archive.com/cryptography@metzdowd.com/msg10142.html"

[11] Satoshi's Fortune lower bound is 100M USD, Available Online,

"https://bitcointalk.org/index.php?topic=175996.0"

[12] The Well Deserved Fortune of Satoshi Nakamoto, Bitcoin creator, Visionary and Genius, Available Online, "https://bitslog.com/2013/04/17/the-well-deserved-fortune-of-satoshi-nakamoto/"

[13] Does Satoshi have a million bitcoin?, Available Online, "https://blog.bitmex.com/satoshis-1-million-bitcoin/"

[14] Bitcoin Developer Reference, Available Online, "https://bitcoin.org/en/developer-reference#getblock"

[15] Slush pool, Available Online, "https://slushpool.com/home/"

[16] Block Timestamp, Available Online, "https://en.bitcoin.it/wiki/Block_timestamp"

A Study of CBDC Model Applicable for the Current Banking Environment

Sajan Maharjan, KyungChan Ko, ChangHoon Kang, JongSoo Woo, James Won-Ki Hong

Department of Computer Science and Engineering, POSTECH
{thesajan, kkc90, chkang, woojs, jwkhong}@postech.ac.kr

Abstract

The emergence of blockchain and cryptocurrency technology has induced significant impact on the financial services area. Governments and banks are keen to utilize this novel technology for their benefits of cost reduction, greater transparency and control. This has resulted in the concept of central bank digital currency (CBDC). While different models of CBDC have been proposed, their implementation hinders the current banking system. In this paper, the authors propose a CBDC model, which can be implemented without affecting the current banking system.

I . INTRODUCTION

The emergence of Bitcoin [1] in 2009 inspired by the financial crisis of 2008 has brought upon significant changes in the financial system and the global economy. Bitcoin, an electronic peer-to-peer payment system, allows payments to be made without the need for a financial intermediary. Henceforth, many alternative coins [2] and blockchain [3] platforms have been proposed and developed to provide novel services in the financial area. Due to the services being offered by these blockchain based financial platforms, there is a growing concern of loss in business or fear of being replaced among existing banks and other financial institutions.

Peeking into the banking business, the cost of production and distribution of fiat cash is expensive. The Bank of Korea reports that it costs twice as much to produce coins worth 10KRW and every year bills worth 20 million KRW are subject to damage while 60 billion KRW is invested in reissuance costs [4]. Other shortcomings of cash are loss, theft, burdensome to hold and lack of transparency (as used in funding of illegal activities, terrorism and corruption).

Besides the rise of cryptocurrencies and blockchain-based financial platforms, among other reasons that has sparked interest in the development of central banks digital currencies(CBDC), is the movement towards a cashless society. There are a plethora of applications and payment services offered by third-party payment service providers which have made peerto-peer payments possible without the use of cash. Service providers such as AliPay [5], WeChat Pay [6], M-Pesa [7], etc., boast large user-bases and are able to harness users' payment data and spending habits. These insights are useful to authorities like central banks and governments to establish financial regulations and control over monetary markets. As-is, central banks lack the aforementioned level of authority in the current cashless society. In response, central banks across the globe are making efforts towards the development of CBDCs as a means of cashless payments aimed at reducing cost of production and circulation, facilitating governance and control along with ease to users.

Unlike cash, which is a physical form of money, a CBDC is an electronic form of money issued by a central bank. Simply defined, CBDCs are monetary values stored electronically (digitally, or as an electronic token) that represent liability of the central bank and can be used to make payments [8]. While there is pre-existing electronic central bank money in the form of settlement balances and reserves, these are only accessible to commercial banks and financial institutions. These may be argued as a form of CBDC but these are not the only exclusive forms of CBDC. Different forms of CBDC may be distinguished from their design choices- level of anonymity, accessibility to the general public, border constraints, token-based vs account-based, etc. While there are different perspectives on CBDC, the most general definition of CBDC was illustrated via a venn-diagram by Bech and Garratt (Fig. 1) [9].

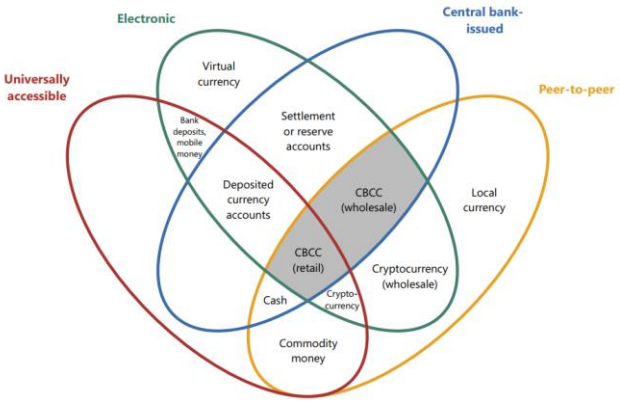


Fig. 1 Money Flower; shaded area represents central bank issued digital currencies.

Based on the above mentioned parameters like accessibility, availability, interest payments, different forms of CBDCs can be defined – Retail CBDC, in which the general public owns a digitized form of central bank money and can be used in retail transactions; wholesale CBDC, in which digital central bank money is issued to banks and financial institutions only for use in large-value transactions; cross-border CBDCs, which facilitates transactions between two countries defined by an exchange rate. Two-tiered CBDCs have also been defined in which commercial banks act as an intermediary to circulate digital currency to the general public. While different forms of CBDCs have been proposed, details on implementation are few.

With the introduction of CBDCs, often discussed is how commercial banks would require to change their business models or even seek other means of making profit. This paper proposes a model of CBDC which is in congruence with the existing banking model. That is, commercial banks can coexist in the CBDC environment without incurring an overhaul of changes. While a small overhead is introduced in the system using the proposed CBDC model, the banking system is benefitted through the added value of blockchain and digital currencies, i.e., less cost of production and circulation, better governance and control. Section II presents related work on CBDC. Section III outlines the design of the proposed model and implementation guidelines. Finally, section IV concludes the paper with potential future work and improvements.

Ⅱ. RELATED WORK

The Bank for International Settlement reports that several nations across the globe have taken an interest in the development, testing, pilot or feasibility analysis of different CBDC projects. Engert and Fung point out the motivation behind CBDCs and their implications on the financial system [8]. The Bank for International Settlements presents a taxonomy on different forms of CBDCs, their design considerations and challenges in their implementation [10]. The International Monetary Fund (IMF) categorizes different forms of money currently in use in the global payment landscape [11]. The IMF argues that e money offered by third party payment service providers are the most widely used forms of payment and that banks must either cooperate or compete with them to dominate the payments market. In the paper, IMF introduces the concept of Synthetic CBDCs whereby central banks and e-money service providers collaborate in a non-zero sum game fashion to increase individual benefits. Additionally, World Economic Forum provides a guideline for policy makers to undertake/deploy a particular form of CBDC given different design requirements [12].

While there are numerous undertakings on surveying the feasibility, application benefits, design considerations and challenges to the development of CBDCs, there are only a handful of papers describing the technical details in implementation. Danezis and Meiklejohn have proposed RSCoin, a cryptocurrency framework in which central banks maintain complete control over the monetary supply, but rely on a distributed set of authorities or mintettes to prevent doublespending [13]. RSCoin builds upon the limitations of Bitcoin – wasteful hashing and lack of governance, and proposes a centralized control of monetary supply henceforth increasing the scalability and stability in the system. Wust et al. have proposed PRCash, a blockchain based currency with central governance which makes use of zero-knowledge proofs and homomorphic encryption to hide the details of transactions [14]. As opposed to RSCoin which focuses on scalability of transaction processing, PRCash focuses on user anonymity while also guaranteeing governance and regulation.

In this paper, we present a CBDC model based on a permissioned blockchain architecture which can be implemented in the current banking environment with only a few changes. The details of the proposed model is discussed in the section below.

Ⅲ. DESIGN & IMPLEMENTATION

A. Requirements

The proposed CBDC model is expected to solve the existing problems of banking environment, i.e., cost of production and circulation of physical cash as well as have features such as better transparency and control. Any digital currency system is able to overcome the inherent problems of fiat cash. Payment service providers like AliPay, WeChat Pay provide direct supply of e-money from supplier to user without intermediaries. However, such payment services could not incorporate the current banking environment where there are intermediaries like commercial banks and financial institutions. The proposed CBDC model must not bring a massive overhaul of the banking system. Commercial banks should not need to seek new avenues of business. Also, the proposed CBDC model should be simple and easy to use (less technically difficult) as people can be reluctant to learn new technology. This can be ensured by user-friendly design of user applications.

We also point out the roles of central banks, commercial banks and users in the system. Central banks will be responsible for the creation of digital currency, enrollment of commercial banks into the CBDC model and distribution of digital currency to commercial banks. Contrary to the existing banking models, central banks will have an added responsibility of registering users under their domain. This is done so as to gain better transparency of users' assets and holdings. Commercial banks will be responsible for enrolling users into their own domain accounts while providing other financial services such as loans, deposits and interest payments. Commercial banks will also need to map users based on their accounts with the accounts held at central banks. General users will be required to enroll under central

bank accounts before registering under commercial bank accounts. General users will also be able to do peer-to-peer transfers via a web or a mobile application without transaction fees.

We also propose a blockchain-based solution compared to centralized database as blockchain based system can providebetter transparency, redundancy and greater system availability. Off-the-shelf enterprise solutions are expensive and not easily customizable. On the other hand, open-source blockchain based payment systems are free to use and can be customized to support financial applications provided by commercial banks (via integration with smart contracts). Central banks will maintain greater control and authority over the blockchain while commercial banks and general users will be appending data to the blockchain when conducting transactions.

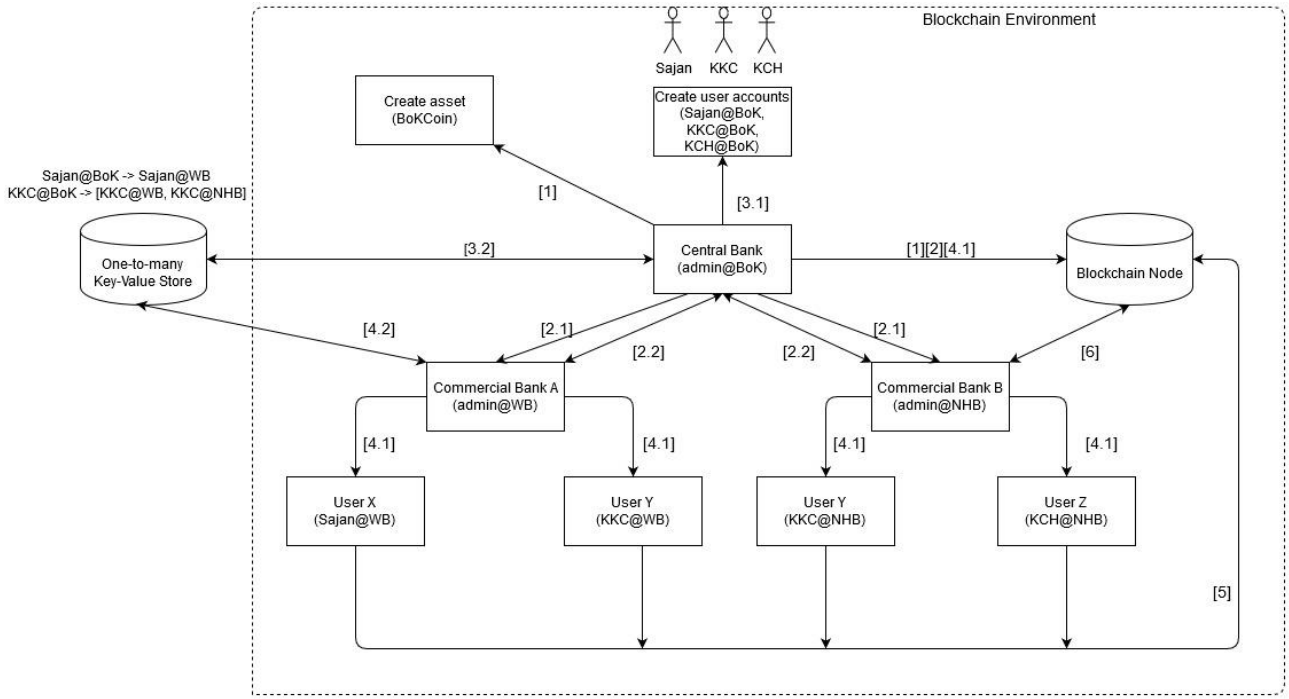


Fig. 2 Proposed CBDC system architecture

B. Design

The overall CBDC architecture shows how different entities in the system interact with the blockchain (Fig 2).

Under the above proposed architecture, the CBDC model is setup as follows-

1 The central bank creates a digital currency for operation in the blockchain platform. Currency issuance is recorded in the blockchain.

2.1 The central bank registers various commercial banks into the blockchain platform. Accounts of commercial banks are written to blockchain.

2.2 The central bank circulates the newly created digital currency to commercial banks. Bidirectional arrow here indicates that commercial banks may pay back the circulated coins at a later time. These transactions are also appended to the blockchain.

3.1 Users interested to participate in the system are only approved after creating accounts with the central bank.

3.2 The central bank records the user details into a key-value store. The key is the account id created at the central bank while the value contains the list of accounts that would be created at other commercial banks. This step enables tracking assets/currency of a user distributed among different banks.

4.1 Users interested in services offered by commercial banks register under accounts of corresponding banks.

4.2 Commercial banks update the key-value store maintained by central banks.

5 Users conduct transactions between themselves (peer-to-peer transfer) and transaction data is recorded in the blockchain.

6 Commercial banks at a later time can conduct different transactions (for example, on behalf of clients, offering of additional services, etc). These transactions are also appended to the blockchain.

The above system architecture depicts the operation under a single node. Multiple nodes can be set up when there are a diverse set of users distributed geographically. In such a scenario, a user can access the nearest node and send transactions via a user interface, for example a web application. The blockchain platform should provide Application Programming Interfaces (APIs) that can be used to interact with the blockchain through the web application (Fig. 3).

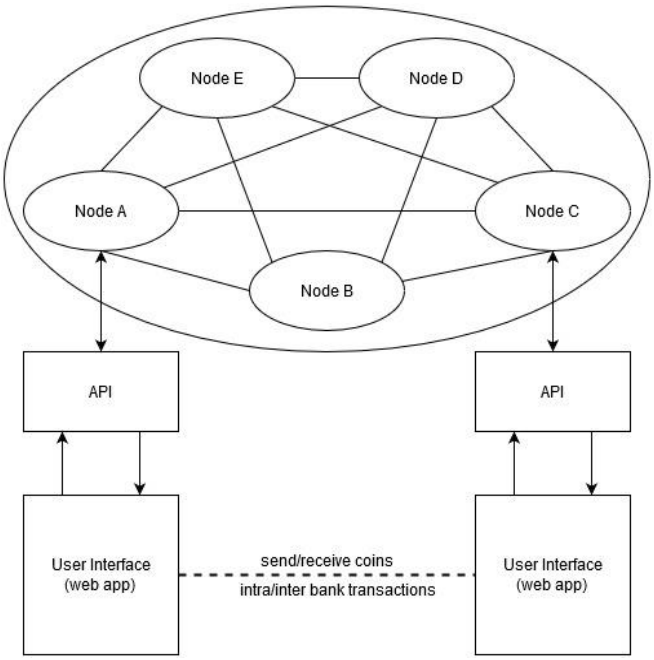


Fig. 3 User interaction mechanism with the blockchain platform in CBDC setting

Traditionally, commercial banks are also responsible for providing additional financial services like loans and deposits, installments, interest payment and other schemes to the general public beside the perfunctory role of currency circulation. The proposed CBDC model considers these services offered by commercial banks through the means of smart contracts [15]. Smart contracts allow automated execution of transactions when certain conditions are met. Financial services such as interest accrued on deposits and loans, tax calculations, installment payments can be smoothly executed via smart contracts. In our model, commercial banks offer these services by deploying smart contracts on top of the blockchain platform (Fig. 4).

Thus, we argue that the proposed CBDC model fits better with the current banking environment than other models of CBDCs that have been suggested. These suggested models argue that the CBDC approach could affect banking business whereby commercial banks are obliged to change their business model and seek other ways of making profit. By using the proposed models, governance and control can be maintained by central banks while still facilitating the commercial banks business models.

C. Implementation Guideline

The development of the proposed model is currently a work-in-progress. The authors suggest that such development be made by using the Hyperledger (HL) [16] blockchain environment. Specifically, under the Hyperledger foundation, the HL Iroha [17] project offers a permissioned blockchain system that can be used to manage digital assets and are used in applications related to interbank settlements, CBDCs, payments systems, etc. The HL Iroha platform can also be integrated with another project under the HL foundation, HL Burrow [18], which allows easy integration of smart contracts on top of HL Iroha. To build the front-end for users, HL iroha offers various software development kits (SDKs) for web and mobile applications for example, Python and Android.

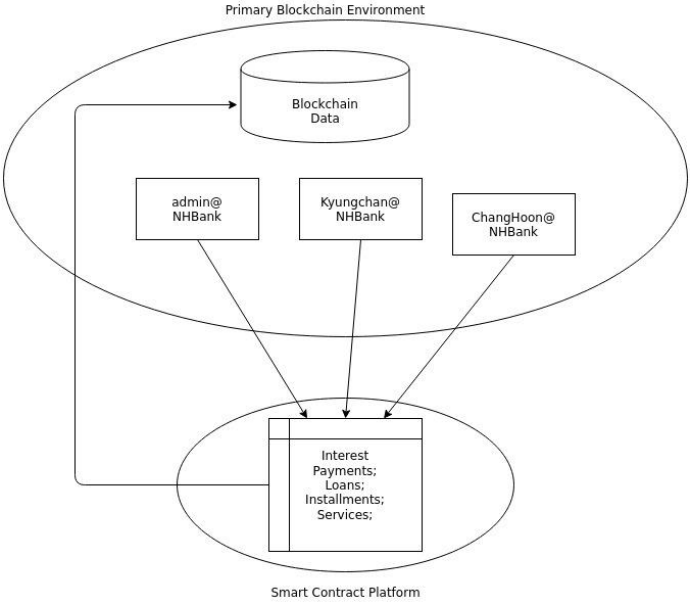


Fig. 4 Services offered by commercial banks are executed by the means of smart contracts

Ⅳ. CONCLUSION

The concept of CBDCs is on the rise in recent years, garnering attention from several governments and central banks. While different models for CBDCs are being proposed and suggested, their implementation is being hindered due to its impact on the current banking systems. In this paper, the authors have suggested an applicable model for CBDC which can be implemented without requiring an overhaul of the current banking system. Central banks will issue coins and govern transactions within the system, while commercial banks will circulate the currency and provide additional services to individuals by the means of smart contracts. Overall, the suggested model of CBDC benefits from the added value of CBDCs (cost of production and distribution, control, transparency) without impacting the current banking system.

The proposed system is currently under development which uses python-sdk for providing user interface through a web application and operating under a single node. When the system needs to be scaled with respect to the high demand from users, web applications could be replaced by providing user interface through mobile applications (android-sdk) and deploying multiple nodes. The authors have only proposed a prototype for the proposed model and thorough testing of system scalability will be done in the near future.

ACKNOWLEDGMENT

This work was supported by the ICT R&D program of MSIT/IITP. [No.2018-0-00539, Development of Blockchain Transaction Monitoring and Analysis Technology]

REFERENCES

[1] S. Nakamoto. "Bitcoin: A peer-to-peer electronic cash system", 2008. (https://bitcoin.org/bitcoin.pdf)

[2] Yuval Gov, "Altcoins - The complete guide". CryptoPotato. (https://cryptopotato.com/altcoins-the-complete-guide/)

[3] Crosby M, Pattanayak P, Verma S, Kalyanaraman V. "Blockchain technology: beyond bitcoin" in Applied Innovation Review Issue 2, June 2016. (https://j2-capital.com/wpcontent/uploads/2017/11/AIR-2016-Blockchain.pdf)

[4] Jin-Bae Kim. "[What is Block] Central Bank issues digital currency, 'CBDC' ". BlockMedia. (https://www.blockmedia.co.kr/archives/114942)

[5] AliPay Inc. "AliPay". (https://intl.alipay.com/)

[6] Tencent. "WeChat Pay". (https://pay.weixin.qq.com/index.php/public/wechatpay_en)

[7] Hughes N. & Lonie S. "M-PESA: Mobile money for the 'unbanked': Turning cellphones into 24-hour tellers in Kenya" in Innovations: Technology, Governance Globalization Issue 2, 2007. (https://www.gsma.com/mobilefordevelopment/wp-content/uploads/2012/06/innovationsarticleonmpesa_0_d_14.pdf)

[8] Walter, E. and Fung, Ben S.C. "Central Bank Digital Currency: Motivations and Implications", Bank of Canada Staff Discussion Paper, 2017. (https://www.bankofcanada.ca/wp-content/uploads/2017/11/sdp2017-16.pdf)

[9] Bech, M and R Garratt. "Central bank cryptocurrencies" in BIS Quarterly Review, Bank for International Settlements, September 2017. (https://www.bis.org/publ/qtrpdf/r_qt1709f.pdf)

[10] Committee on Payments and Market Infrastructure. "Central Bank Digital Currencies", Bank for International Settlements, March 2018. (https://www.bis.org/cpmi/publ/d174.pdf)

[11] Adrian, T. & Mancini-Griffoli, T. "The Rise of Digital Money. Fintech Notes", International Monetary Fund. 2019. (https://www.imf.org/en/Publications/fintech-notes/Issues/2019/07/12/The-Rise-of-Digital-Money-47097)

[12] Center for the Fourth Industrial Revolution. "Central Bank Digital Currency Policy-Maker Toolkit", World Economic Forum. 2020. (http://www3.weforum.org/docs/WEF_CBDC_Policymaker_Toolkit.pdf)

[13] George Danezis and Sarah Meiklejohn. 2016. "Centrally Banked Cryptocurrencies" in 23nd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016. (https://arxiv.org/pdf/1505.06895.pdf)

[14] Karl Wust, Kari Kostiainen, Vedran Capkun, and Srdjan Capkun. "PRCash: Fast, Private and Regulated Transactions for Digital Currencies." In Financial Cryptography and Data Security - 23rd International Conference, FC 2019. (https://doi.org/10.1007/978-3-030-32101-7_11)

[15] Nick Szabo. "Smart Contracts: Building blocks for digital markets". 1996. (http://www.truevaluemetrics.org/DBpdfs/BlockChain/Nick-Szabo-Smart-Contracts-Building-Blocks-for-Digital-Markets-1996-14591.pdf)

[16] The Hyperledger Foundation, "Hyperledger". (https://www.hyperledger.org/)

[17] The Hyperedger Foundation, "Hyperledger Iroha". (https://www.hyperledger.org/projects/iroha)

[18] The Hyperledger Foundation, "Hyperledger Burrow". (https://www.hyperledger.org/projects/hyperledger-burrow)

Distributed SDN Based Network State Aware Architecture for Flying Ad-Hoc Network (FANET)

Muhammad Saqib*, Afaq Muhammad, Wang-Cheol SONG
Jeju National University Jeju-si, Jeju-do, South Korea

*saqib@jejunu.ac.kr, afaq@jejunu.ac.kr, kingiron@gmail.com

Abstract

Due to resources constraints, high mobility, multi-interconnections and self-organizing nature, flying networks are very prone to underutilization of resources, unpredicted link failure and performance degradation. A resource and fault aware distributed network architecture is proposed for Flying Ad- hoc Network that early predict network failure by continuously monitoring the network state information, and adjust the near future changes in an automated way.

Ⅰ. Introduction

In aerospace networks, however, the properties of the candidate network topology change constantly due to highly dynamic nature of nodes. UAV networks pose special characteristics of high dynamics, unstable aerial wireless links and UAV collision probabilities. While it is possible to treat such changes the same way as traditional failures, e.g., by building redundancy into the network and/or reactively repairing breakages when they occur, doing so leads to inefficient use of resources and to user visible network disruptions.

The inherit challenge is to ensure network availability by considering the challenges for proactive routing, radio spectrum and wireless channel utilization. It would be on- ly possible by developing a centralized network resources aware system that can pro- actively configure the network based on the global network knowledge.

Software defined network (SDN) architecture offers new opportunities to introduce innovative applications and incorporate automatic and adaptive control aspects, there- by easing network management and proactively mitigating the effects of network events [1]. Despite the excitement, SDN adoption raises many challenges including the scalability and reliability issues of centralized designs that has been addressed with the physical decentralization of the control plane [2]. However, such physically distributed, but logically centralized systems bring an additional set of challenges. This has led to categorization of existing controller platforms into centralized and distributed architectures [3].

This document is categorized in 2 sections. Section 2 introduce the research goal and objectives. Section 3 covers the proposed architecture in details with each of the service modules. Section 3 concludes the proposals aspects.

Ⅱ. Research Goal & Objectives

The overall goal is to achieve network availability and scalability by designing SDN based resources and fault aware distributed network architecture for Flying Ad-hoc Network. Following are the objectives:

- To incorporates nodes' mobility information and wireless channel properties to predict network failure and proactively adjust the network topology in anticipation of future changes
- To improve network performance and resource allocation based on the global network knowledge.
- To improve the network scalability and reduce control traffic overhead by physically distributing control plane
- The complete synchronization cannot be used in our scenario as we also have to maintain best network performance. We can use partial synchronization for inter-controller updates.
- To offer fast link recovery in case of unpredicted link failure by embedding sufficient intelligence inside

network nodes (embedded reactive distributed routing protocol.

Ⅱ. Proposed Architecture

For effective state of subject, it is necessary to develop a scalable, resources and network state aware architecture that can timely predict the network failure in highly dynamic networks. There are several ways to achieve this task. For example, the UAVs mobility information and wireless channel properties can be monitored by centralized SDN controller with optimal decision. The networks resources may have utilized based on the global network knowledge which may lead to improved network performance. But a single SDN controller can be a single point of failure and the control traffic overhead may degrade the controller performance. Therefore, in our research, the control plane has to be distributed. In distributed control plane, each controller has to independently control its local domain, and may share its local domain knowledge with neighbors' controllers to form a wide network view. Each controller may adequately intelligent and autonomous decision maker for its local domain, to proactively configure the network structure based on early failure prediction. Moreover, the network nodes may also embed with sufficient intelligence to automatically follow alternate path in case of unpredicted link failure.

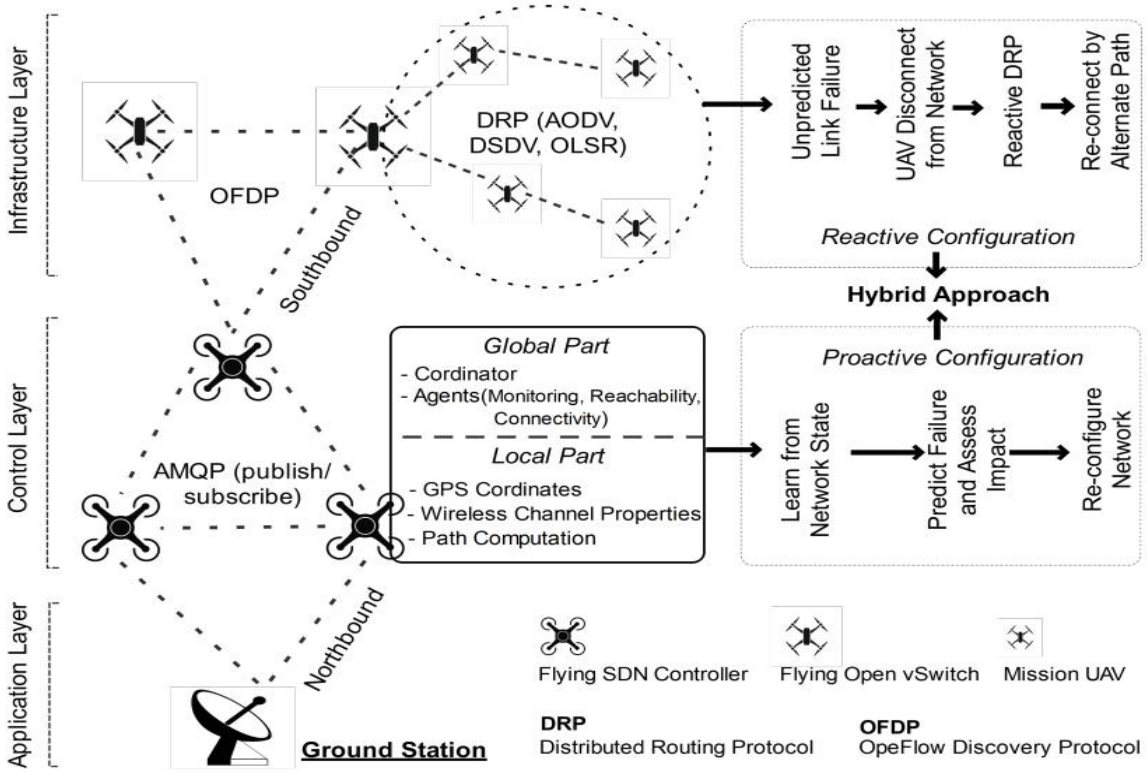


Fig: Proposed Architecture Diagram

In our methodogly, we should follow these steps:

- SDN Controller: Firstly, the SDN controller platform is important to select for system development. For example, POX is a Python based SDN controller platform.
- Controller Distribution: We will extend the existing controller functionalities by dividing into two parts: local and global. Local part would be responsible for network state monitoring and autonomous decision making for its local domain. Whereas, global part may responsible for communicating and sharing local network domain knowledge with neighbor controllers. Most importantly, the selection of suitable protocol and mechanism may necessary for communication between distributed controllers. For instance, AMQP is publish/subscribe protocol.
- Network Failure Prediction: We will develop a failure prediction system on top of the SDN controller that can monitor wireless communication channels, get

the localization of UAVs, access the characteristic of UAV movement and forecast the network structure changes.

- Unpredicted Link Failure: We will embed sufficient intelligence (reactive distributed routing protocol) into mission UAVs to establish connectivity in case of unpredicted link failure. For example, AODV, DSDV and OLSR are the well most widely used distributed routing protocols.

As our work is based on distributed architecture, all the controllers would have to proactively share their local topology updates in a timely fashion to avoid routing misbehaviors (e.g. loops). Based on network state information, the domain controller can proactively assign weight to each link to indicate the link preference. The self-domain synchronization (each controller only knows its own intra-domain topology with its link weights) can be used for intra-domain shortest path finding based link weights. For inter-domain synchronization, the partial synchronization can be used to reduce the synchronization cost by dividing the traversing domains into routing clusters and synchronize the bordering domains only instead of complete synchronization. The complete synchronization cannot be used in our scenario as we also have to maintain best network performance. Moreover, we also have to use simple path design to not use the repeated vertex among domain controllers.

Ⅲ. Conclusion

This work proposed a state of the art highly desirable solution for highly dynamic networks where nodes appear and disappear very simultaneously. By incorporating the real-time network state information, the network resources can be efficiently utilized which can lead to better network availability. Moreover, the scalability can also be achieved by distributing the control, where the controller monitors their local domain and share aggregated network knowledge to form a global network view.

References

[1]. Iqbal, H., et al. A software-defined networking architecture for aerial network optimization. in 2016 IEEE NetSoft Conference and Workshops (NetSoft). 2016. IEEE.

[2]. Elzain, H. and W. Yang. Decentralizing Software-Defined Wireless Mesh Networking (D- SDWMN) Control Plane. in Proceedings of the World Congress on Engineering. 2018.

[3]. Bannour, F., S. Souihi, and A. Mellouk, Distributed SDN control: Survey, taxonomy, and challenges. IEEE Communications Surveys & Tutorials, 2018. 20(1): p. 333-354.

기계 학습 기반 VNF 관리 시스템 제안

김희곤, 이도영, Stanislav Lange, 유재형, 홍원기
포항공과대학교 컴퓨터공학과,

{sinjint, dylee90, stasl, jihyoo78, jwkhong}@postech.ac.kr

The VNF Management System using Machine Learning

Hee Gon Kim, Do Yeong Lee, Stanislav Lange, Jae Hyung Yoo, James Won Ki Hong
Department of Computer Science and Engineering, POSTECH

*Graduate school of Information Technology, POSTECH

요 약

본 논문은 기계 학습 기반의 VNF 관리 시스템을 제안하였다. 가상 네트워크 환경에서 기존의 관리 시스템은 네트워크의 동적인 변화에 빠르게 대응을 할 수 없으며 최적의 관리가 불가능하였는데, 이러한 문제점을 기계 학습 기반의 VNF 관리 시스템이 해결 할 수 있음을 서술하였다. 또한 단순하게 추상적인 시스템 구조만을 표현한 것이 아니라 기존 연구들의 한계점을 지적하면서 구체적으로 어떠한 기계 학습 모델이 VNF 를 관리하는 데 적합한 지 설명하였다. 본 논문에서는 Graph Neural Network (GNN)를 사용하여 네트워크를 학습하는 것을 제안하였으며, 학습에 필요한 데이터 및 전체 VNF 관리 시스템 구조를 서술하였다.

Ⅰ. 서 론

Software-Defined Networking (SDN) 과 Network Function Virtualization (NFV) 는 OPEX 와 CAPEX 를 감소시키며 네트워크를 동적이고 유연하게 만들어 준다. 기존의 네트워크는 실제 물리 네트워크 장치들이 연결된 구조를 가지고 있었으나 SDN 과 NFV 환경에서는 범용 서버들이 네트워크를 구성하고 있으며, 각 서버 위에 설치된 가상 네트워크 기능(Virtual Network Function: VNF) 들이 물리 네트워크 장치 기능들을 대신 수행한다. 이러한 네트워크 구조의 변화는 기존의 네트워크보다 빠르게 환경을 변화시킬 수 있는 장점을 가지는데, 한편으로는 네트워크의 복잡도와 동적 변화를 증가시켜 관리의 난이도를 증가시키기도 한다.

전통적으로 네트워크 관리 방법은 Integer Linear Programming (ILP) 을 기반으로 네트워크 환경을 수학적으로 계산하여 최적의 환경을 구성하였다. 하지만 SDN 과 NFV 도입으로 네트워크 환경들은 점점 더 복잡해졌고, ILP 계산의 소요시간이 길어지게 되었다. 또한 네트워크의 동적 변화가 자주 일어남에 따라 ILP 로 구한 관리 정책이 더 이상 변화된 네트워크에서 최적의 정책임을 보장할 수가 없게 되었다. 이러한 문제를 해결하기 위해 최근에는 빠른 계산 시간을 보장하는 기계학습 기반 네트워크 관리 연구가 주목을 받고 있다.

기계학습 기반의 네트워크 관리의 목표는 사람의 개입 없이 스스로 관리를 하는 자율 네트워크 시스템의 구축이다. 현재 이와 관련한 연구로 [1]은 기계 학습을 사용하여 Virtual Network Embedding 문제를 해결하고자 하였으며 [2]는 VNF Deployment 문제를 여러 기계학습 모델을 사용하여 해결하였다. 하지만 이러한 연구들이 실제 네트워크에 적용할 수 있냐고 질문한다면 그렇다고 대답하기는 어렵다. [1]와 [2]와 같이 많은 연구들[3][4]은 기계 학습을 네트워크 환경에 적용하고자 노력을 하고 있기는 하지만 그들이 제안한 방법을 살펴보면 오직 한정된 환경(네트워크의 구조 변화가 없음)과 정해진 조건(요구되는 네트워크 서비스의 종류 및 양의 변화가 없음) 그리고 간단한 네트워크 정책(특정 VNF 의 개수 관리 등)에 대해서만 행해진 것을 확인 할 수 있다.

본 논문은 VNF Migration 과 VNF deployment 모두를 고려하는 넓은 의미의 뜻에서 VNF 관리를 실시하며, 네트워크 구조가 고정되지 않고 동적으로 변화하는 환경에서 변화하는 네트워크 서비스 요구에 따라 최적의 관리 방법을 기계 학습이 계산하는 시스템을 제안한다. 또한 이러한 시스템 구현을 위해 우리는 기계 학습의 일종인 Graph Neural Network (GNN)[5]을 사용하는 것을 제안한다.

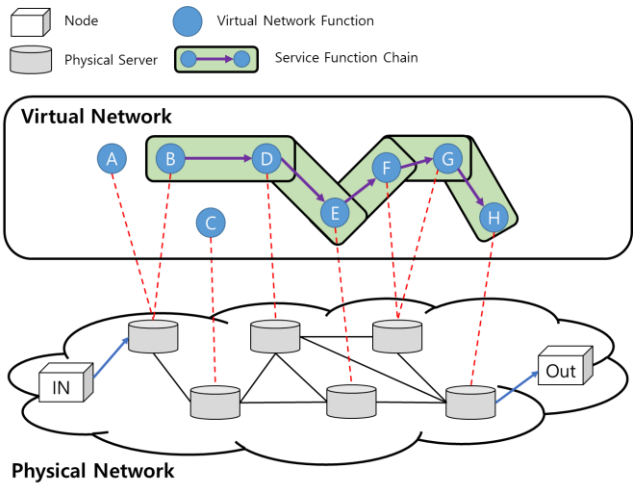


그림 1 물리 네트워크와 가상 네트워크 개념도

Ⅱ. 본론

그림 1 은 물리네트워크와 가상네트워크의 개념을 간단히 나타낸 것이다. VNF 는 네트워크의 물리 서버 위에서 동작하며 사용자가 특정한 네트워크 서비스를 요구하면 그 서비스를 충족할 수 있는 VNF 들이 순차적으로 동작한다. 서비스를 제공하기 위해 동작하는 VNF 의 열들은 Service Function Chain (SFC) 라고 불리며 그림에서는 초록색으로 표현되었다. 네트워크 관리는 네트워크 서비스가 요구될 때 SFC 를 고려하여 VNF 최적의 위치로 Deploy 혹은 Migration 하는 것이며, 이때 네트워크 자원 사용을 최적화하면서 지연이나 실패 없이 서비스를 제공하는 것이다. 예시로 그림 1 을 살펴보면, 현재 서비스는 B->D->E->F->G->H 의 VNF 체인을 가지는 SFC 를 통하여 제공되고 있는데, VNF E 와 VNF F 가 위치한 물리 서버들 간에는 직접적인 연결이 존재하지 않기 때문에 VNF E 에서 VNF F 로 가기위해서는 다른 서버를 거쳐야만 연결이 되는 것을 확인 할 수 있다. 이러한 경우에는 VNF E 를 VNF F 가 위치한 물리 서버로 Migration 하거나 Deploy 하는 방법으로 문제를 해결할 수 있다[6]. 또한 이러한 Migration 과 Deploy 정책 결정사항 시 대상이 되는 물리 서버와 전체 네트워크의 가용 자원이 해당 정책을 실시하기에 충분하며, 결정된 정책이 전체 네트워크 자원을 최적으로 사용하는 지 파악하여야 한다. 즉 네트워크 관리는 사용자의 서비스 요구 사항 정보와 네트워크의 구조 및 자원 상태 정보를 필요로 한다.

네트워크 관리에 있어 필요한 정보를 구체적으로 나타내면 다음과 같이 정의를 할 수 있다.

- 서비스 정보
  - 요구되고 있는 서비스의 종류
  - 요구되고 있는 서비스의 양
  - 요구되고 있는 서비스의 SLA
  - 서비스를 요구하는 노드 및 제공하는 노드의 위치
- 네트워크 정보
  - 물리 네트워크 토폴로지 연결 형태
  - 물리 서버 간의 Latency 및 Bandwidth
  - 물리 서버의 가용 자원
  - 물리 서버의 VNF 종류
  - VNF 의 가용 자원

본 논문에서는 위에서 정의된 서비스 정보와 네트워크 정보를 사용하여 VNF 관리하는 네트워크 관리 시스템을 제안한다. 전통적인 네트워크 관리 시스템에서는 ILP 를 사용하여 문제를 해결하였지만, 복잡한 네트워크에서 위의 정의된 모든 정보를 사용하여 빠르게 관리 방법을 찾는 것은 거의 불가능에 가까우며, 정답을 찾은 경우에는 이미 서비스 정보와 네트워크 정보가 변화한 상태에 직면하게 된다. 이러한 문제를 해결하기 위해 본 논문은 기계 학습을 사용하는 것으로 관리 정책을 결정하는 시스템을 제안하였다.

기계 학습을 사용하여 네트워크를 관리하려는 시도는 새로운 것은 아니다. 하지만 기존의 시도들은 특정한 네트워크 환경에서만 관리 모델이 작동한다는 고질적인 문제점을 벗어나지 못했다. 기계 학습은 데이터를 사용하여 학습하고 학습 결과를 바탕으로 주어진 문제를 푸는 모델이다. 결국 기계 학습은 주어진 데이터에 종속적이게 되는 데, 기계 학습 기반 네트워크 관리 모델은 학습에 사용된 네트워크 환경에 종속되게 된다. 따라서 많은 기계 학습 기반 네트워크 연구들은 오직 한정된 환경과 목적에서만 정상적으로 작동을 하게 된다. 특히 치명적인 것은 네트워크는 주로 동적으로 계속 변화하기 때문에, 현재 모델이 정상적으로 작동하고 있다 할지라도 언제 모델이 정상적으로 작동하지 않을지 모른다는 것이다.

기계 학습 모델이 한정된 데이터에 종속적이라면 여러 다양한 환경의 네트워크 데이터를 학습하는 것으로 모델이 범용성을 갖추게 할 수도 있다. 하지만 기존의 기계 학습 기반 네트워크 관리 모델은 다양한 환경의 네트워크 데이터를 사용하고 싶어도 학습 모델의 성능 저하로 인해 데이터를 사용을 할 수 없었다. 다양한 네트워크 환경의 데이터를 적용할 수 없는 이유는 네트워크 데이터가 가지고 있는 고유한 특징때문이다. 기계 학습 모델은 일련의 간단한 벡터 형태의 숫자 데이터를 입력으로 받고 결과 데이터로 다시 숫자 데이터를 생성하는데, 네트워크 데이터를 입력 데이터로 사용하기 위해서는 먼저 네트워크 정보를 일련의 숫자 데이터로 표현을 해야한다. 이때 앞서 정의했던 서비스 정보와 네트워크 정보를 숫자로 표현한다고 할 때, 서비스의 양, SLA, Latency, Bandwidth 의 경우는 직접적인 수치로 간단히 표현이 가능하다. 또한 요구되고 있는 서비스의 종류나 물리 서버의 VNF 종류와 같은 정보도 one-hot-encoding 이나 embedding vector 를 사용하는 것으로 일련의 숫자로 분류되는 표현이 가능하다[7]. 하지만 물리 네트워크 토폴로지 연결 형태와 같은 정보는 표현이 어렵다. 네트워크의 각 물리 서버들 간의 연결 정보를 나타내기 위해서는 앞선 SLA 나 VNF 종류와는 달리 고차원의 벡터가 필요하며 복잡한 형태의 데이터로 표현이 된다. 이러한 문제로 기존의 연구들은 네트워크 토폴로지 구조를 학습 데이터에서 제외하였다. 그런데 이렇게 토폴로지 구조가 학습 데이터에서 제외되면, 학습 모델은 여러 다양한 환경의 네트워크 데이터를 학습 할 때 토폴로지를 구분할 수 없어 혼란을 겪게 되고 제대로 된 관리 정책을 내릴 수 없게 된다. 즉 모델은 토폴로지 정보를 가지지 않기 때문에 학습에 사용된 네트워크 데이터가 Star-토폴로지 네트워크인지 Link-토폴로지 네트워크인지, 물리 서버 간의 링크가 복잡하게 많이 형성된 네트워크인지, 간단한 네트워크인지 구분할 수 가 없게 된다.

본 논문은 네트워크 학습 모델이 범용성을 갖출 수 있도록 Graph Neural Network (GNN)을 사용하는 것을 제안한다. GNN 은 그래프 형태의 데이터를 입력으로 받아 그래프를 더 적은 형태의 그래프로 압축하거나 더 큰 형태의 그래프로 생성할 수 있으며 그래프 전체를 하나의 특정한 형태의 정보로 표현을 할 수 있다. 우리는 전체 네트워크 정보를 그래프 형태 데이터로 나타내어 GNN 에 입력 데이터로 사용을 하였는데, 네트워크

토폴로지 구조 자체가 그래프 형태의 데이터이기 때문에, 기존의 기계 학습 모델과는 달리 학습 모델이 네트워크 토폴로지 구조를 학습할 수 있게 되었다. 그래프로 표현된 네트워크 데이터는 GNN 학습 이후 일련의 숫자 데이터로 표현이 되고, 우리는 이렇게 표현된 숫자 데이터를 다시 서비스 정보 데이터와 합쳐 일반적인 기계 학습 모델 방법인 Feed Forward Neural Network (FNN) 을 사용하는 것으로 최적의 관리 모델을 찾을 수 있게 된다. GNN 을 사용하는 것으로 얻을 수 있는 또 하나의 장점은 네트워크 그래프 전체에 대해 학습이 이루어지기 때문에, 하나의 개별 물리 서버 만을 대상으로 VNF Deployment 최적화 정책만을 얻는 것이 아니라 전체 네트워크의 모든 환경을 고려하여 모든 물리 서버를 대상으로 VNF Deployment 정책을 얻으며 물리 서버 간의 VNF Migration 정책을 얻을 수도 있다는 점이다. 즉 제안하는 기계 학습 방법은 서비스 정보와 네트워크 정보를 고려하여 모든 VNF 에 대해서 어느 위치에 어떠한 VNF 어떠한 VNF 관리 정책을 따라야하는 지 학습하는 것이 모두 가능하다.

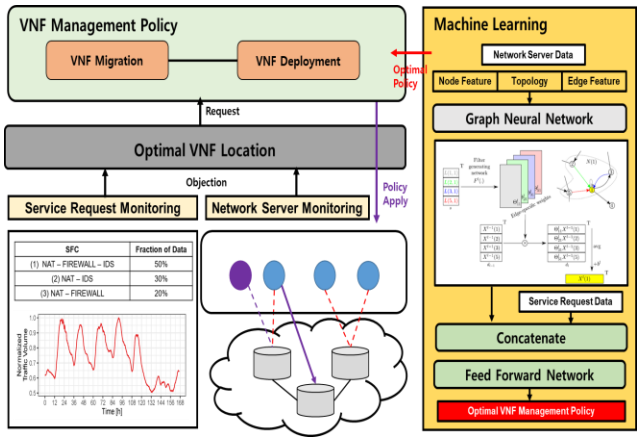


그림 2 기계학습 기반 VNF 관리 시스템

그림 2 는 제안한 기계학습 기반 VNF 관리 시스템을 간단히 도식화 한 것이다. 시스템은 VNF 들이 최적의 위치에 설치되는 것을 목표로 하면서 서비스 정보와 네트워크 정보를 모니터링하고 수집한다. 시스템의 목표는 VNF Migration 과 VNF Deployment 정책을 요구하며, 요구된 정책은 기계 학습을 통해 결정이 된다. 기계 학습은 그래프 형태로 표현된 네트워크 정보 데이터를 GNN 을 사용하여 일련의 숫자 벡터로 학습시킨 이후, 서비스 정보를 Concatenate 하여 FNN 에 학습 시킨다.

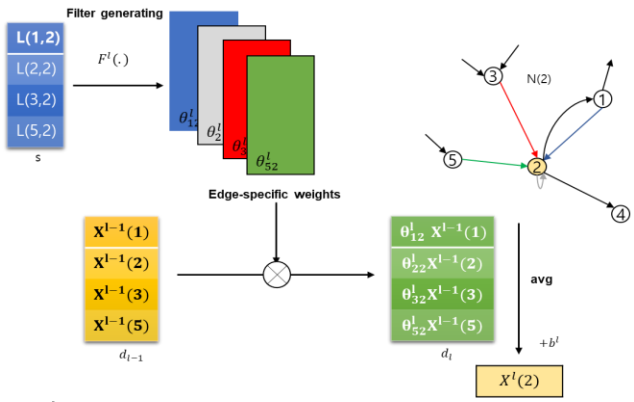


그림 3. Edge-conditioned Filters in Convolutional Neural Network

III. Edge-conditioned Filters in Convolutional Neural Network

본 논문은 GNN 모델 중 Edge-conditioned Filters in Convolutional Neural Network[8] 를 사용하는 것을 제안한다. Edge-conditioned Filters in Convolutional Neural Network 는 그래프의 노드와 엣지가 가지고 있는 정보를 활용하여 새로운 정보를 생성한다. 그림 3 은 이 GNN 모델이 어떻게 이웃 노드와 엣지를 이용하여 학습을 하는 지 표현하고 있다. 다섯 개의 노드가 존재하는 그래프를 주고, 노드 정보를 X, 엣지 정보를 L 라고 정의했을 때, 2 번 노드와 연결된 엣지들은 FNN 을 통해 엣지 정보의 특징을 가지고 있는 필터(Θ)로 변환이 되며, 필터는 2 번 노드를 포함한 이웃 노드 정보와 곱셈을 하고 평균이 되어 새로운 정보가 된다.

이러한 일련의 과정은 노드 정보가 이웃 노드 정보와 엣지 정보를 포함하게 하는 효과가 있다. 네트워크에서 노드를 물리서버라고 정의를 하고 엣지를 노드 간의 연결이라고 정의를 한다면, 원래 노드들은 물리 서버의 가용 자원과 같이 오직 하나의 물리 서버 정보만을 가지고 있지만 그림 3 의 과정을 진행하게 되면, 노드 정보가 이웃 노드, 즉 이웃 물리 서버의 정보와 서버 간의 링크 정보를 가지게 된다. 모든 노드에 대해서 그림 3 의 과정을 진행하면 노드들의 정보만으로도 엣지 정보와 그래프 전체 정보를 알 수 있게 된다. 즉 GNN 을 사용하면 노드와 엣지로 구성된 고차원의 그래프 형태의 데이터를 노드 단위의 기계 학습이 쉽게 이해할 수 있는 데이터로 변환을 할 수 있다.

IV. 결론

본 논문에서는 기계학습 기반 VNF 관리 시스템을 제안하였다. 제안한 시스템은 GNN 을 이용하여 동적으로 변화하는 네트워크를 학습하며, 최적의 VNF 위치를 찾을 수 있다. 제안한 모델은 기존의 연구와 달리 네트워크 데이터를 그래프 형식의 데이터로 사용하여 네트워크의 동적 변화를 모델이 학습할 수 있으며 학습 결과로 어떠한 VNF 가 네트워크의 어느 물리 서버에 위에 Deploy 되거나 제거되어야 하는 지 구체적인 관리 정책을 제공할 수 있다. 본 논문에서는 구체적인 모델과 학습에 필요한 데이터를 제안하였으며, 제안한 모델을 구현하는 것으로 인공지능 기반의 자율 VNF 관리 시스템을 개발할 수 있을 것이다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획 평가원의 대학 ICT 연구센터지원사업의 연구결과로 수행되었음 (IITP-2020-2017-0-01633)

참 고 문 헌

[1] Blenk, Andreas, et al. "Boost online virtual network embedding: Using neural networks for admission control." *2016 12th International*

Conference on Network and Service Management (CNSM). IEEE, 2016.

[2] Lange, Stanislav, et al. "Predicting VNF Deployment Decisions under Dynamically Changing Network Conditions." *2019 15th International Conference on Network and Service Management (CNSM)*. IEEE, 2019.

[3] Rahman, Sabidur, et al. "Auto-scaling vnfs using machine learning to improve qos and reduce cost." *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018.

[4] Jeong, Seyeon, et al. "Machine Learning based Link State Aware Service Function Chaining." *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2019.

[5] Gori, Marco, Gabriele Monfardini, and Franco Scarselli. "A new model for learning in graph domains." *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*. Vol. 2. IEEE, 2005.

[6] Bari, Md Faizul, et al. "On orchestrating virtual network functions." *2015 11th International Conference on Network and Service Management (CNSM)*. IEEE, 2015.

[7] Rodríguez, Pau, et al. "Beyond one-hot encoding: Lower dimensional target embedding." *Image and Vision Computing* 75 (2018): 21-31.

[8] Simonovsky, Martin, and Nikos Komodakis. "Dynamic edge-conditioned filters in convolutional neural networks on graphs." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

네트워크 운용 자동화를 위한 SDN 기술 연구 및 적용사례

김준혁, 김경열, 김우태
KT 융합기술원

jh1227.kim@kt.com, kyoungyoul.kim@kt.com, utae.kim@kt.com

A Study and Use Case on the SDN Technology for Network Operation Automation

Kim Jun Hyuk, Kim Kyoung Youl, Kim Woo Tae
KT Institute of Convergence Technology

요 약

Software-Defined Network(이하 SDN)기술은 네트워크 사업자에서 주목 받고 있으며, 5G 인프라 구축 시 네트워크 기능 가상화(NFV) 기술과 함께 사용되고 있는 기술이다. 일반적으로 SDN 기술은 물리적인 네트워크 장비에서 동작되는 컨트롤 평면과 데이터 평면을 분리해 소프트웨어처럼 사용자가 원하는 기능을 자유롭게 구현하여, 네트워크 장비 벤더 종속성 없이 네트워크의 운용과 관리를 유연하게 하고 확장성 있는 인프라를 구축하는데 목적이 있다.[1] 컨트롤 기능은 중앙 집중화된 컨트롤러가 수행하며, 각 지역에 분산 구축되어 있는 스위치는 컨트롤러의 플로우 룰 실행을 통해 동작이 되며, OpenFlow 가 주요기술로 많은 관심과 표준화가 이루어졌지만, 최근에는 ONF(Open Networking Foundation)가 주도하는 P4(Programming Protocol-independent Packet Processor) 기술 기반 SDN 이 주목을 받고 있다.[2]

본 논문에서는, 상기 기술을 활용하여 네트워크 인프라 운용 관점에서의 SDN 기술과 개념 적용을 통해 통합, 중앙 집중형 자동화 기술 및 사례에 대한 소개를 하고자 한다.

Ⅰ. 서 론

SDN 은 네트워크 분야에서 최근 몇 년간 주목 받는 연구분야 중 하나로써, 하나의 물리적 네트워크 장비에서 동작되는 컨트롤 평면과 데이터 평면을 분리해 소프트웨어처럼 사용자가 원하는 기능을 자유롭게 구현하여 네트워크의 운용과 관리를 유연하게 하고 확장성 있는 인프라를 구축하는데 목적이 있다.

컨트롤 기능은 중앙 집중화된 컨트롤러가 수행하며, 각 지역에 분산 구축되어 있는 라우터와 스위치는 컨트롤러의 플로우 룰 실행을 통해 동작이 되며, 초기에는 OpenFlow 가 주요기술로서 많은 관심과 표준화가 이루어졌다. 이러한 SDN 의 주요목적 구현을 위해서는 Network Functions Virtualization(NFV) 기술과 같이 결합되어 추진 되어야 하며, 국내 및 글로벌 통신사업자에서도 5G, IoT 등의 분야에서 SDN/NFV 기술을 결합한 PoC 및 상용화를 추진하였다.[3]

본 논문에서는 네트워크 인프라 운용 관점에서 SDN 기술과 SDN 도입을 통한 중앙 집중형 통합 자동화 기술 및 사례에 대한 소개를 하고자 한다.

Ⅱ. 네트워크 인프라 및 관리 프로토콜

SDN 자동제어 체계 도입을 위해서는 네트워크 인프라 운용 관점 및 관리 프로토콜 관점에서 검토가 되어야 한다. 먼저, 위 두 가지 관점에서 네트워크 현황을 살펴보기로 한다.

첫째, 인터넷 서비스 제공을 위한 네트워크 인프라는 End-to-End Connectivity 제공을 위한 IP 계층, 원거리 연결, 파장분할 등을 위한 전송계층, 그리고, 실제적인 광케이블 연결을 위한 OSP(Out-Side Plant) 계층으로 구분이 된다.

현재 대부분 사업자들은 각 계층 별 관리 및 운용이 독립적으로 이루어지고 있으며, KT 는 네트워크 인프라 중 전송계층 자동화를 위한 솔루션으로, 2016 년 세계최초 T-SDN 을 개발하여 성공적 사업화를 추진 하였다.[3][9] SDN 기반 네트워크 전 계층 자동화를 위해서는 계층 상호간의 연결성을 위한 검토와 통합하여 제어를 할 수 있는 오케스트레이터가 필수적이다.

그리고, 네트워크 운용은 지역분산형 구조로, 운용자들이 CLI(Command Line Interface)를 통해 설정 및 운용을 하고 있다.

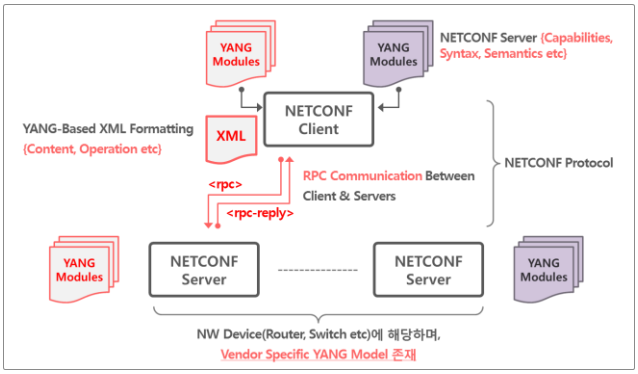
이러한, 지역분산형 운용자 중심의 환경에서는 명령어 오입력에 따른 휴먼에러 발생 위험이 항상 존재한다. 운용장비의 기종 별, 장비 별 네트워크 설정이 상이하여 운용자의 조기기량 확보에 한계가 존재하는 문제점이 있으며, 휴먼에러의 큰 원인 중의 하나이다.

특히, 5G 시대 네트워크 기능 가상화(NFV) 기술을 통한 네트워크 인프라 운용 복잡성은 더 복잡해 질 수 있으므로 SDN 중심의 자동화 기술을 통한 운용은 필수적이다.

둘째, 네트워크 자동화를 위해 네트워크 인프라 현황과 업무분석 못지 않게 네트워크 관리 프로토콜 역할이 중요하다. 대부분의 네트워크 사업자들이 사용하는 장비는 P4 기능을 지원하는 장비가 아닌 레거시 네트워크 장비로 99% 이상 차지하고 있다. 따라서, 기존 레거시 시스템에서 SDN 기반 네트워크 자동화를 위해 적용 가능한 네트워크 관리 프로토콜이 매우 중요하다고 할 수 있다.

2002 년 IAB(Internet Architecture Board) 워크숍에서 기존 네트워크 운용관리를 위한 자동화 기술에 대해 심도 있는 검토를 하였으며[5], 2003 년 IETF NETCONF 워킹그룹을 출범하여, NETCONF 외 네트워크 데이터 모델링을 위한 YANG 표준화를 추진하였다.[6][7]

네트워크 설정관리를 위해 표준화한 NETCONF/YANG 프로토콜은 2010 년 SDN 에 대한 관심이 높아지면서, SDN 자동화 기술로 주목을 끌기 시작하였다.



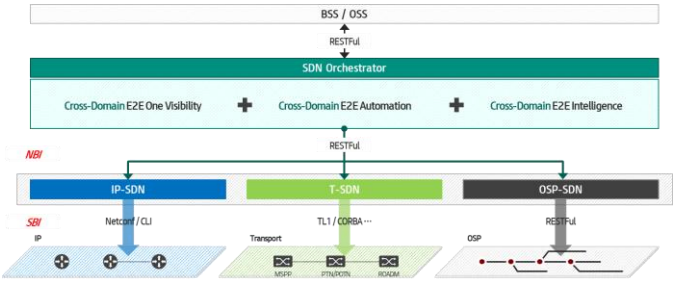
그러나, 표준화를 추진하였지만, 네트워크 데이터 모델링이 네트워크 장비 제조사 별로 상이하여, 최근에는 네트워크 제조사 별 의존성 없는 모델링인 OpenConfig 표준화를 추진하고 있지만[8] 아직은 성숙되지 않은 기술로 본 논문에서는 언급을 제외하며, NETCONF/YANG 표준 프로토콜 기반 SDN 통합 제어 기술을 소개하고자 한다.

Ⅲ. SDN 통합제어 구조

Ⅱ절에서 언급한 바와 같이, 인터넷 서비스 제공을 위한 네트워크 인프라는 IP 계층, 전송계층 그리고 OSP 계층으로 이루어져 있다. SDN 기반 통합 제어를

위해서는 계층간 상호연결성 및 제어를 위한, 각 계층의 SDN 컨트롤러와 인프라 오케스터레이터의 역할이 매우 중요하다.

각 계층의 SDN 은 IP-SDN, Transport-SDN 과 OSP-SDN 으로 구성이 되며, 각 SDN 컨트롤러에서 수집한 정보를 이용하여 상호연결성 정보 자동확보, 통합자동 제어 기반 인프라를 확보할 수 있게 된다.



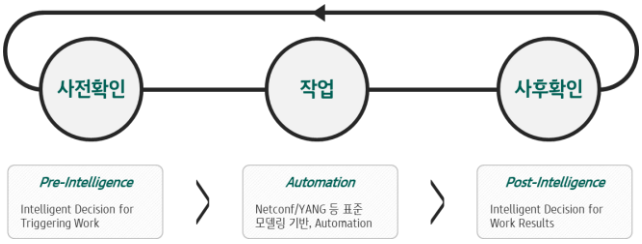
상기 SDN 오케스터레이터는 각 사업자의 BSS/OSS 연동을 통하여, 서비스 및 네트워크 구성 등 네트워크 운용 업무에 대해 SDN 기반 통합 자동화 제어를 이룰 수 있게 된다.

그리고, 각 계층의 SDN 컨트롤러는 데이터 모델링 기반의 RESTful, RESTCONF 등 NBI(North Bound Interface)와, 실제 네트워크 디바이스 자동설정 등을 위한 SBI(South Bound Interface)로 구성이 된다.

Ⅳ. 사례 분석

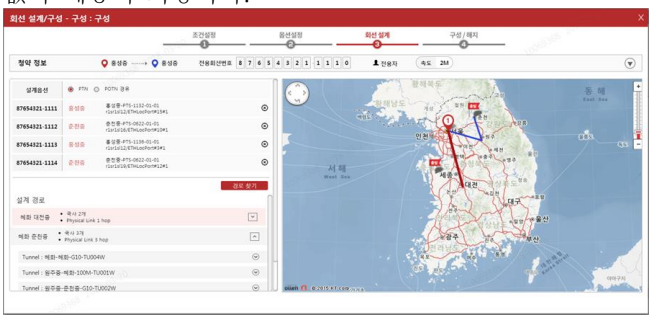
본 절에서는 실제 네트워크 운용 중 자주 발생하는 업무의 SDN 기반 End-to-End 자동화에 대해 보기로 한다.

네트워크 오퍼레이터의 운용업무는 운용자의 작업에 대한 사전 확인단계 - 작업 - 결과 모니터링 과정을 거치게 된다. SDN 기반 자동화에서는 운용자의 판단이 필요한 사전 확인단계(Pre-Intelligence)와 사후 모니터링 단계(Post-Intelligence)는 지능화에 의해 시스템이 판단을 하며, 작업단계는 NETCONF/YANG 과 같은 네트워크 설정 프로토콜을 통해 작업(Automation)이 이루어지게 된다.

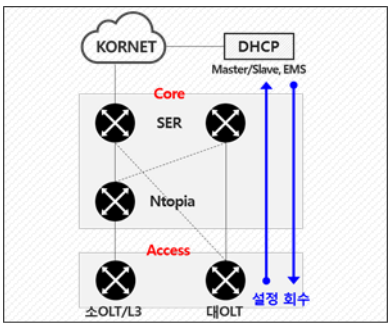


SDN 적용 사례는 전송회선 구성 및 End-to-End IP 관리(설정/회수) 자동화 경우에 대해 보고자 한다. 본 업무 자동화를 통해 휴먼에러 제로화는 물론, 작업시간도 1/10 이상 절감하여 운용 효율성을 극대화 하였다.

전송회선은 T-SDN 을 통해 End-to-End 회선구성을 위한 조건/옵션(E-Line/Tree/LAN 등) 설정 – 회선설계 – 구성단계를 거쳐 End-to-End 자동화 구성이 원클릭으로 완료가 된다. 그리고, 장비대개체 등 절체가 필요한 경우의 전체 회선도 원클릭으로 서비스 중단 없이 제공이 가능하다.



IP 계층에서는 IP 설정 등의 관리 작업이 빈번하게 일어나며, 한 장비에서만 설정하면 완료되는 것이 아니라 DHCP 서버에서 L3 스위치 구간까지 모두 구성이 되어야 한다. 특히, IP 주소 경우는 한 비트의 값만 잘못 설정되어도 인터넷 서비스에 영향이 있으므로 정확하게, End-to-End 로 실수 없이 설정하는 것이 매우 중요하며, 이러한 모든 기능이 IP-SDN 을 통해 자동으로 이루어진다.



그리고, SDN 을 통해 네트워크 구성업무 자동화에 적용이 되지만, 네트워크 사업자는 시설, 형상, 연결정보의 관리 또한 매우 중요하다. 이러한 네트워크 장비에 대한 관리도 SDN 을 통해 자동으로 정확하게 이루어질 수 있다.

V 결론

본 논문에서는 효율적인 네트워크 운용을 위해 SDN 기반 중앙 집중형 통합 자동 제어 구조 및 사례에 대해 살펴 보았다.

네트워크 사업자는 다양한 유무선 기술을 통한 인프라가 구축되는 환경에서, 네트워크 운용업무 효율성을 가지기 위해 SDN 기반 통합 자동제어 체계 확보가 필수적이며, 많은 네트워크 사업자들이 SDN 자동화를 위해 다양한 시도를 하고 있다.

마지막으로, 네트워크 운용의 통합 자동제어를 위해서는 자동화와 지능화가 동시에 이루어져야 네트워크 운용 효율화 관점에서 의미가 있다. 현 단계에서는 네트워크 운용업무 규칙 기반 지능화이지만, 딥 러닝 기반 인공지능(AI)을 통한 지능화 기술이 미래 네트워크 인프라 운용의 핵심 기술이 될 것이며, 통신사업자들은 관련된 기술개발에 집중하여 다양한 분야에서 PoC 및 상용화를 추진하고 있다. 빠른 시일 내 네트워크 자동화와 지능화 기술을 통해 네트워크 사업자의 운용생산성을 향상시켜 사용자에게도 안정적이고 고품질의 인터넷 서비스가 제공 가능할 것으로 기대한다.

참 고 문 헌

[1] 유재형, 김우성, 윤찬형, “SDN/OpenFlow 기술동향 및 전망”(2015, KNOM Review)
[2] P4 https://www.opennetworking.org/p4/
[3] KT “통신사업자 관점의 5G 네트워크 기술(2016)”
[4] ZDNet Korea “KT, T-SDN 전국 상용화 완료”(2016)
[5] RFC3535, “Overview of the 2002 IAB Network Management Workshop”
[6] RFC4741, “NETCONF Configuration Protocol”
[7] RFC6020, “YANG – A Data Modeling Language for the NETCONF”
[8] Draft-openconfig-netmod-model-structure (2015)
[9] Chosun Biz “KT, SDN 기반 지능형 네트워킹 자동 솔루션 개발”(2017)

NFV 관리를 위한 VNF 이상 탐지 시스템에 대한 연구

홍지범, 박수현, 유재형, 홍원기

포항공과대학교 컴퓨터공학과

{hosewq, sh.park11, jhyoo78, jwkhong}@postech.ac.kr

A Study on VNF Anomaly Detection System for NFV Environment Management

Jibum Hong, Suhyun Park, Jae-Hyoung Yoo, James Won-Ki Hong

Department of Computer Science and Engineering, POSTECH

요 약

Software-Defined Networking (SDN) 및 Network Function Virtualization (NFV)의 개념이 제안된 이후, 현재 통신 사업자 및 서비스 제공업체는 SDN/NFV 기술을 활용하여 기존의 서비스를 보다 효율적으로 제공하기 위해 노력하고 있다. 하지만 데이터 센터에서 운용되는 가상 네트워크가 점점 복잡해짐에 따라 리소스 할당, 장애 관리 등과 같은 다양하고 새로운 네트워크 관리 문제가 발생하게 되었다. 복잡해지는 관리 문제를 해결하기 위해서는 우선 가상 네트워크에서 동작하는 Virtualized Network Function (VNF)의 리소스 사용량 및 네트워크 트래픽 로드를 모니터링하고 분석할 필요가 있다. 이를 위해 본 논문에서는 NFV 환경에서 시스템 리소스 및 트래픽 과부하로 인해 발생하는 Service Level Agreement (SLA) 위반과 관련된 VNF 의 이상 상태를 탐지하는 시스템을 제안한다. 제안하는 시스템은 실제 테스트베드에서 수집한 데이터로 머신러닝 (Machine Learning) 모델을 학습시켜 가상 네트워크 환경에서 동작하는 VNF 들의 이상 상태 (anomaly) 탐지에 활용할 수 있다.

Ⅰ. 서 론

오늘날 Software-Defined Networking (SDN) 및 Network Function Virtualization (NFV) 기술은 네트워크 가상화를 통해 네트워크를 구축하고 운영하는 새로운 방법을 제공하고 있다. SDN/NFV 기술은 기존 하드웨어 중심의 폐쇄된 네트워크 기능을 소프트웨어 형태의 Virtualized Network Functions (VNFs)로 대체하여 비용을 절감시킨다. 또한 클라우드 컴퓨팅 (Cloud Computing)은 이러한 기술들을 활용하여 데이터 센터에서 소요되는 컴퓨팅 리소스를 보다 효율적으로 사용하고, 다양한 애플리케이션 서비스를 유연하고 효율적으로 배포 하는 것을 가능하게 한다 [1].

이에 따라 통신 사업자와 서비스 제공업체는 최근 SDN/NFV 및 클라우드 컴퓨팅 기술을 이용하여 다양한 애플리케이션 서비스를 운영하거나 가상 환경에서 Evolved Packet Core (EPC), IP Multimedia Subsystem (IMS)과 같은 핵심 네트워크 기능을 구현하고 있다. 이를 위해 통신 사업자 및 서비스 제공업체는 가상 네트워크에 가상 머신 (Virtual Machine) 또는 컨테이너 (container)를 설치한 후, 원하는 서비스를 동작시킨다.

하지만 기존 하드웨어 기반의 네트워크 장치 운용과는 다르게 NFV 환경에서는 가상 서버 또는 가상 네트워크처럼 자원들을 가상화 하여 서비스 운용이 이루어지기 때문에 가상 네트워크가 점점 복잡해진다는 단점이 존재한다. 이로 인해 리소스 할당 최적화, 장애 관리 등과 같은 다양하고 새로운 네트워크 관리 문제가 발생하게 되어 서비스 및 시스템의 심각한 장애를 유발시킬 수 있다. 따라서 가상 네트워크에서 동작하는 VNF 의 시스템 리소스 사용량 및 네트워크 트래픽 로드와 같은 동작 상태를 분석할 필요가 있다.

본 논문에서는 가상 네트워크의 관리 문제를 해결하기 위한 방법의 일환으로, 머신러닝 (Machine Learning)을 통해 Service Level Agreement (SLA) 위반과 관련된 지표를 학습시켜 서비스를 제공하는 VNF 의 이상 상태를 실시간으로 탐지 (Anomaly Detection)하는 시스템을 제안한다. 그리고 제안하는 이상 탐지 시스템을 활용한 향후 연구 방향을 제시한다.

Ⅱ. 관련 연구

VNF 의 이상 탐지는 CPU, 메모리 사용량과 같은 측정치 (metric)의 임계값 (threshold)을 이용하여 탐지할 수 있지만 이는 단순히 한 측정치의 임계값을 기반으로 이상 여부를 판단하기 때문에 많은 오탐 (false alarm)을 유발한다. 따라서 기존의 이상 탐지 연구는 대체로 시계열 (time-series) 데이터 및 통계 알고리즘을 이용하여 VNF 및 시스템의 상태를 판단하고 있다. STL (Seasonal Trend Decomposition using LOESS) 알고리즘을 적용하여 이상 상태를 탐지하는 연구 [2]는 클라우드 환경에서 수집한 시계열 CPU 데이터의 계절성 요인을 함께 고려하여 3-Sigma 규칙에 의해 정의된 임계값을 넘는 측정치를 이상 상태로 탐지한다.

최근에는 머신러닝과 같은 인공지능 기술을 네트워크 관리에 적용하려는 연구가 활발히 진행되고 있다. 선행연구 [3]는 vIMS 환경에서 동작하는 컨테이너화된 (containerized) VNF 의 이상 상태를 지도학습 (supervised learning) 기반의 Random Forest 알고리즘

을 이용하여 탐지하는 방법을 제안하고 있다. 또한 다른 머신러닝 알고리즘을 적용한 선행 연구 [4]에서는 Support Vector Machine (SVM), Decision Tree (DT), Neural Network (NN) 알고리즘을 통해 VNF 의 성능 이상을 탐지하고 각 학습 모델의 성능을 비교하고 있다.

Ⅲ. 머신러닝 기반의 VNF 이상 탐지 시스템

기존 VNF 이상 탐지 시스템은 대부분 CPU 과부하, 메모리 부족 등의 상태만을 이상으로 판단한다는 한계가 존재한다. 본 논문에서는 리소스 및 네트워크 사용량을 포함하여 SLA 위반과 관련된 지표를 기반으로 서비스의 상태를 반영함으로써 보다 정확하게 이상 상태를 탐지하는 시스템을 제안한다. 제안하는 머신러닝 기반의 VNF 이상 탐지 시스템의 구조는 그림 1 과 같다.

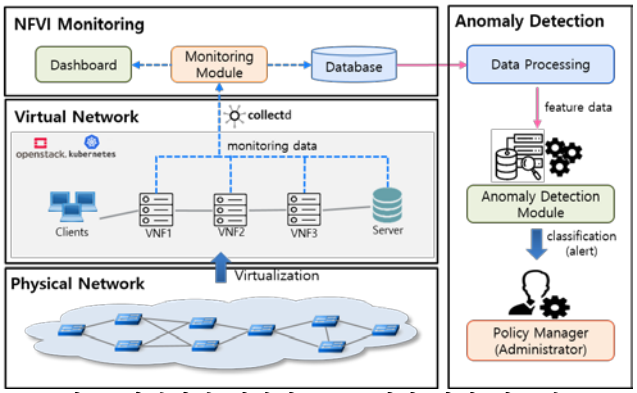


그림 1 머신러닝 기반의 VNF 이상 탐지 시스템 구조

먼저 하드웨어를 통해 구성된 물리 네트워크 환경에서 OpenStack 및 Docker/Kubernetes 와 같은 Virtual Infrastructure Manager (VIM)을 이용하여 NFV Infrastructure (NFVI)를 구축한다. 이러한 NFVI 환경은 인공지능 기반의 NFV 관리 플랫폼 선행 연구 [5]를 기반으로 이루어지며, 다양한 서비스를 제공하기 위해 구성된 가상 네트워크에서 각 서비스에 부합하는 VNF 들을 동작시킨다. 운영중인 가상 네트워크에 대한 데이터는 각 VNF 내부에 설치된 모니터링 에이전트인 Collectd 에서 CPU, memory, disk I/O 와 같은 리소스 사용량, 네트워크 트래픽 로드 등과 같은 네트워크 데이터를 실시간으로 수집하여 모니터링 모듈로 보낸다. 모니터링 모듈은 전달받은 데이터를 대시보드에 전달하여 실시간으로 모니터링 데이터를 시각화하여 보여주거나 데이터베이스 플랫폼에 보내어 시계열 형태의 데이터로 저장한다.

다음으로 머신러닝을 통해 사전 학습하여 도출된 모델 기반의 이상 탐지 모듈은 데이터베이스에서 주기적으로 데이터를 가져온 후, VNF 의 이상 상태 탐지를 위한 분류에 사용하기 위해 데이터를 feature 데이터로 변환시킨다. 이렇게 변환된 feature 데이터를 기반으로 이상 탐지 모듈은 실시간으로 가상 네트워크에서 동작 중인 VNF 들의 현재 상태를 정상 및 이상 상태로 분류하여 판단한다. 이 때 이상 탐지 모듈이 특정 VNF 의 상태에 문제가 있다고 판단을 하면 이상 탐지 모듈은 NFV 환경 관리를 위한 정책을 세우는 Policy Manager 혹은 네트워크 관리자에게 경보 (alert)를 보내어 운영 중인 서비스에 이상 징후가 발생했음을 알린다. 이와 같은 머신러닝 기반의 VNF 이상 탐지 시스템을 구축하기 위해 머신러닝 알고리즘을 통해 이상 탐지 모델을 학습시키는 과정이 필요하다. 가상 네트워크에서 수집한 데이터를 통해 VNF 이상 탐지 모델을 학습시키는 과정은 그림 2 와 같다.

NFVI 환경에서 동작 중인 가상 네트워크를 모니터링하여 데이터를 수집 및 저장하는 방법은 앞선 방법과 동

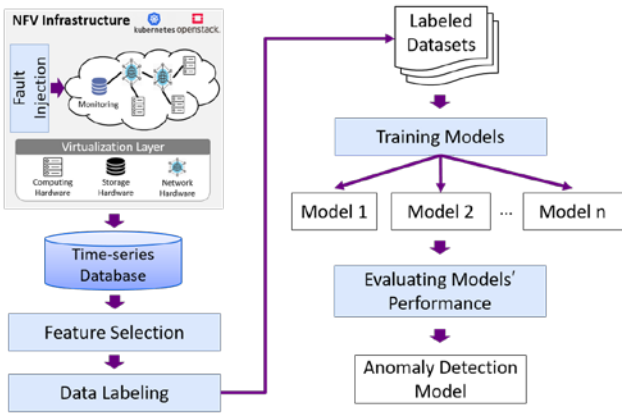


그림 2 VNF 이상 탐지 모델의 학습 과정

일하며, 이를 통해 CPU, memory, disk I/O, 트래픽 로드 등과 같은 다양한 측정치를 세분화하여 수집한다. 하지만 가상 네트워크의 운영 데이터를 수집할 때 실제 네트워크 환경에서는 비정상적인 동작을 나타내는 이상 상태가 거의 발생하지 않으므로 결함 주입 (fault injection)을 통해 CPU 과부하, 메모리 부족, 트래픽 과부하 등의 이상 상태를 발생시켜 SLA 를 위반하도록 유도한다.

이렇게 수집한 모니터링 데이터는 시계열 데이터베이스에 저장된다. 저장된 데이터는 지도학습 기반의 머신러닝 알고리즘 학습에 활용할 수 있도록 수집한 데이터의 여러 측정치 중에서 이상 상태 분류에 핵심이 되는 feature 를 추출하며, 데이터 레이블링은 결함 주입으로 패킷 손실률 (packet loss rate), 지연 (latency), 서비스 요청 실패율 (service request failure rate) 등과 같은 SLA 위반이 발생했을 시점의 데이터를 이상 상태로 레이블링하여 정상 및 이상 상태의 데이터셋을 생성한다.

다음으로 레이블링된 데이터셋을 통해 지도학습 기반의 머신러닝 알고리즘을 각각 학습시킨다. 학습을 위한 환경은 머신러닝/딥러닝 프레임워크인 TensorFlow 및 Pytorch 를 이용한다. 학습에 사용되는 알고리즘은 앞서 관련 연구에서 언급한 SVM, DT, Random Forest 와 같은 알고리즘과 함께 Gradient Boosting Machine (GBM), eXtreme Gradient Boost (XGBoost) 등과 같이 최근 뛰어난 성능을 보여주는 알고리즘을 사용하여 학습시킨다.

마지막으로, 각 알고리즘들을 기반으로 학습된 이상 탐지 모델들의 성능을 accuracy, F1-Score 등을 이용하여 비교한 후, 가장 좋은 성능을 보이는 모델을 제안하는 시스템에서 채용하는 이상 탐지 모듈의 기반 모델로 도출하여 이상 탐지 시스템을 구현한다.

Ⅳ. 결론 및 향후 연구

본 논문에서는 NFV 환경 관리를 위한 머신러닝 기반의 VNF 이상 탐지 시스템을 제안하였다. 제안하는 방법은 가상 네트워크 환경에서 데이터를 수집하여 지도학습 기반의 머신러닝 알고리즘을 통해 이상 탐지 모델을 학습시키고, 이를 SLA 위반 여부를 바탕으로 제공 중인 서비스의 이상 징후를 탐지함으로써 기존 이상 탐지 시스템을 개선하여 서비스의 심각한 장애가 발생하기 전에 네트워크를 선제적으로 관리할 수 있다.

향후 연구로 제안하는 방법을 바탕으로 가상 네트워크의 데이터를 생성 및 수집한 후, 다양한 머신러닝 알고리즘의 성능을 비교하여 최적의 성능을 보이는 VNF 이상 탐지 모델을 도출한다. 또한 도출된 모델을 바탕으로 VNF 이상 탐지 시스템을 구현한다. 마지막으로, 실제 환경에서의 실험을 통해 제안하는 이상 탐지 시스템의 성능을 검증한다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2015-0-00575, 글로벌 SDN/NFV 공개소프트웨어 핵심 모듈/기능 개발, 2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발).

참 고 문 헌

[1] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "Openstack: toward an open-source solution for cloud computing," International Journal of Computer Applications, vol. 55, no. 3, pp. 38-42, Oct. 2012.

[2] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," Computing Research Repository, abs/1704.07706, 2017.

[3] C. Sauvanaud, K. Lazri, M. Kaaniche, and K. Kanoun, "Anomaly detection and root cause localization in virtual network functions," In 2016 IEEE 27th International Symposium on Software Reliability Engineering, pp. 196-206, Oct. 2016.

[4] J. Qiu, et al., "Performance anomaly detection models of virtual machines for network function virtualization infrastructure with machine learning," In Artificial Neural Networks and Machine Learning - ICANN 2018, pp. 479-488. Springer International Publishing, Oct. 2018.

[5] 정세연, 이도영, 유재형, 홍원기, "인공지능 기반 NFV 관리 플랫폼," In KNOM Conference 2019, pp. 40-42, May 2019.

비트코인 노드 간 압축 블록 전달 성능 측정 및 분석

이기영*, 주홍택

계명대학교

*1112536@stu.kmu.ac.kr, juht@kmu.ac.kr

Compact Block Relay Performance Measurement and Analysis between Bitcoin Nodes

Kiyoung Lee*, Hongtaek Ju

*Keimyung Univ., Keimyung Univ.

요 약

본 논문은 비트코인 블록체인 네트워크에서 한 노드가 한 개의 블록을 전달받는데 소요되는 시간을 측정하고 분석한 결과를 제시한다. 비트코인 네트워크는 블록이 마이닝 되어 생성되면 소문에 의한 플러딩 방식으로 전체 네트워크에 전파되며 이때 각 노드는 여러 개의 메시지를 교환하여 다음 노드에 블록을 전달한다. 비트코인의 전체 블록 전파 성능은 노드 사이의 전달 성능에 의존적이며 빠른 노드 간 전달은 빠른 블록 전파로 귀결된다. 비트코인의 블록 전달 방식은 최근에 압축 블록 전달(CBR: Compact Block Relay)을 도입하여 성능을 개선하였다. 이 논문에서는 이 방식의 블록 전달 성능을 메시지 단위로 측정하고 분석하였으며 이 결과는 향후 추가적인 블록 전달 성능 개선의 기초자료로 활용될 수 있다. 노드 간 블록 전달 시간을 측정하기 위하여 두 개 비트코인 노드를 직접 연결하고 전달 과정에서 발생하는 메시지 단위의 교환 과정을 밀리 초 단위로 시간을 측정하였다. 측정한 결과 블록 전달 시간은 정규분포를 가지고 있었으며, 블록 전파를 위한 단계를 구간으로 지정하여 시간을 측정하였다. 블록에는 존재하지만, 메모리 풀에 존재하지 않아 재요청한 트랜잭션의 수를 측정하고 분석하였다.

Ⅰ. 서 론

본 논문에서는 사토시 나가모토의 백서로부터 시작된 블록체인 기반의 비트코인 네트워크의 블록 전달 시간을 측정하고 분석한 결과를 제시한다. 비트코인은 P2P 가상 네트워크를 구성하여 노드들이 연결되며 모든 노드에서 동일한 블록체인 데이터를 공유한다[1]. 동일한 블록체인 데이터를 공유하기 위해 블록은 특정 노드에서 마이닝 되어 생성되며 전체 네트워크에 전파 되고 검증을 한 후 저장된다[2].

블록 전파 과정에서 블록이 노드 간 전달되는 시간을 측정하기 위해 비트코인 블록이 생성 될 때 블록에 기록되는 블록 생성 시간을 사용하여 측정하는 방법을 고려할 수 있다. 하지만 이 방법은 두 가지 문제를 가지고 있다. 하나는 블록에 기록되는 블록 생성 시간이 정확하지 못하다. 이때 사용하는 시간은 시스템 시간이 아니라 네트워크 시간으로 마이닝 노드가 연결한 주변 노드의 시간 차이를 고려한 평균값으로 계산된 정확한 값이 아니다[3]. 두 번째 문제는 블록이 생성되어 자신에게 전파 될 때까지의 시간은 측정할 수 있으나 노드 간 전달 시간을 측정할 수 없다. 이 논문에서는 두 노드 간 블록이 전달되는 과정을 측정하고 분석한 결과를 제시한다.

두 노드 간 블록 전파시간을 측정하기 위한 방법으로 패킷을 수집하여 분석하는 방법이 있다. 대표적인 패킷 분석 도구로 잘 알려진 Wireshark를 활용하는 방법으로 Auqib 등이 이 방법을 사용한 연구 결과를 제시하고 있다[4]. 이 연구는 최근 적용된 압축 블록 전달(CBR: Compact Block Relay) 방식을 적용하지 않았고 교환되는 메시지 값만을 제시하여 시간적인 분석을 실시하지 않았다. Ryunokuke 등은 CBR 방식의 성능 개선 결과를 시뮬레이션으로 제시하고 있으나 전달과정에서 메시지 단위의 소요되는 시간을 제시하지 못하였고 실제 시간을 측정하지 못하였다[5]. 기존 연구에서 노드 간 전달 시간을 메시지 단위로 전달 시간을 직접 측정하여 분석한 연구결과는 없다.

이 논문에서는 두 노드를 직접 연결하고 노드 간 블록 전달 시간을 메시지 단위로 측정하기 위해서, 노드에서 실행되는 프로그램이 만들어 내는 로그를 기반으로 시간을 기록하고 분석하였다. 하나의 노드는 블록체인 네트워크에 연결하고 다른 노드는 이 노드에 직접 연결하였으며 두 노드는 한 개 서버에 도커(Docker)로 실행하여 정확한 시스템 시간 측정이 가능도록 하였다. 메시지 교환 시간뿐만 아니라 메시지 교환 중간에 노드가 수행하는 작업을 로그 분석을 통하여 알 수 있으며 이를 통해 어떤 작업에 시간이 많이 소요되는지 분석 할 수 있었다.

블록이 전파되는 과정을 분석하여 각 단계별로 소요되는 시간을 측정하였고 비교하기 위해 그래프로 표현하였다. 그리고 전달 받은 노드가 블록 헤더만을 전달 받은 후 전체 블록을 구성하기 위하여 필요한 트랜잭션을 전달한 노드에게 재요청하는 트랜잭션의 수를 측정하고 분석하였다.

분석한 결과는 상관 계수를 계산한 결과 음(-)의 값으로 측정되고 이 계수의 의미는 블록 전파 시간은 트랜잭션 재요청 횟수가 적을수록 짧게 측정되는 결과를 보여준다.

본 논문에서 제시한 블록 전달 시간 측정 및 분석 결과는 향후 블록 전달 시간 성능 향상을 위한 기초자료로 활용되며 전달 과정에서 발생하는 보안 문제나 이 보안 문제의 해결 방안 도출에도 활용될 수 있다. 보안 문제의 종류에는 이중 지불 공격[6]이나 이기적인 마이닝[7] 그리고 분산 서비스 거부(DDoS) 공격[8] 등이 있다. 대부분의 공격 방식이 비트코인 네트워크 문제로 발생 할 수 있다.

논문은 이후 2장에서 이론적 배경과 관련 연구에 대하여 소개하며 3장에서는 실험 환경 및 수집 데이터에 대하여 다룬다. 4장에서는 분석 결과

를 제시하고 5장은 결론이다.

Ⅱ. 관련 연구

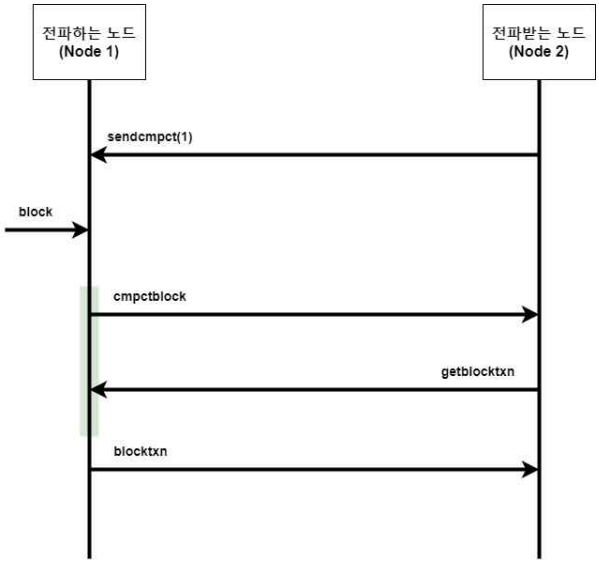
비트코인은 코인의 지불 관계를 기록한 트랜잭션이 지속적으로 발생되며 전체 네트워크에 전파되고 각 노드는 전파된 트랜잭션을 메모리 풀에 저장한다. 전파된 트랜잭션 중에서 일부를 선택하여 마이닝 노드는 블록을 생성하며 이 블록은 10분 주기로 생성되고 전체 네트워크에 다시 전파된다. 블록이 각 노드에 전파되면 각 노드는 블록을 검증하고 이 블록을 블록체인에 연결한다.

블록 전파 과정에서 노드 간 블록이 전달될 때 2개 프로토콜이 사용된다. 하나는 최초에 만들어진 기존 프로토콜과 압축 블록 전달 프로토콜(CBR)이다[9]. 기존 프로토콜은 블록을 노드 간에 전달할 때 블록에 포함된 트랜잭션도 같이 전달하며 통상적으로 블록 당 1MB 정도 트래픽이 발생한다. 기존 프로토콜은 전달 할 노드가 inv (Inventory) 메시지를 전달받을 노드에 보내서 새로운 블록이 있음을 알리고 이에 대한 응답으로 전달 받을 노드는 getdata(Get Data) 메시지로 전체 블록을 요청하고 block(Block) 메시지로 블록을 전달 받는다.

CBR은 전달 받을 노드가 전달 할 노드에게 sendcmpct(Send Compact Block) 메시지를 보내서 CBR로 전달 받고 싶다고 알려야 한다. 이때 2가지 전달 방식이 있는데 하나는 저 대역폭 전달(Low Bandwidth Relay)과 고 대역폭 전달(High Bandwidth Relay)이다. 2가지 방식 모두 전체 블록을 전달하는 대신에 블록헤더와 이 블록에 포함된 트랜잭션의 축소된 ID 값을 전달한다. 전달 받은 노드는 축소된 ID를 사용하여 메모리 풀에서 블록에 포함된 트랜잭션을 찾아 전달 받은 블록 헤더와 함께 블록을 재구성한다. 이렇게 함으로써 블록에 포함된 트랜잭션을 전달 할 필요가 없기 때문에 효율적으로 블록을 전달한다. 물론 전달 받은 노드의 메모리 풀에 해당 트랜잭션이 없으면 getblocktx(Get Block Transaction) 메시지와 blocktxn(Block Transaction) 메시지를 교환하여 부족한 트랜잭션을 가져온다. 저 대역폭 전달은 inv 메시지와 getdata 메시지를 사용하는 것은 동일하나 Block 메시지 대신에 cmpctblock 메시지를 사용한다. 고 대역폭 전달 방식은 inv 메시지와 getdata 메시지를 교환하지 않고 cmpctblock 메시지를 전달 할 노드가 바로 전달 받을 노드에 전달한다. 특히 고 대역폭 전달 방식은 블록을 검증하는 과정과 블록 전달 과정을 동시에 실행하여 네트워크 전파 시간을 단축하였다.[10][11] <그림 1>은 고 대역폭 전달 과정을 보여 주고 있으며 본 연구에서는 이 방식에서의 전달 성능을 측정하였다.

Ryunosuke 등은 압축 블록 전달 방식의 성능 개선 효과를 시뮬레이션을 통하여 검증하였다[5]. 9000개의 노드로 비트코인 네트워크를 구성하고 현실과 비슷한 네트워크 구성을 하였으며 전체 노드 중 97%가 CBR 전달 방식을 사용한다고 가정하였으며 압축 블록의 크기를 18KB로 설정하였다. 시뮬레이션 결과 블록이 전체 네트워크의 50%에 전달될 때까지 걸리는 시간은 CBR이 사용되기 전의 2015년의 경우와 비교하여 9.673초에서 1.340초로 단축됨을 알아내었고 90%에 전달될 때까지는 14.056초에서 2.346초로 단축됨을 알았다. 이 연구는 노드의 전달 시간을 측정하지 않고 전체 네트워크에서 블록이 전파되는 시간을 고려하였다. 본 연구는 전파 시간보다는 전달 시간에 초점을 맞추고 있다.

Auqib 등은 Wireshark 트래픽 분석 도구를 사용하여 블록체인 메시지의 분석 결과를 제시하였다[4]. 이들은 비트코인 프로토콜 규격에 따라서 실제로 많이 사용하고 있는 비트코인 코어 프로그램이 실제로 어떻게 실행되고 있는지 규명하였다. 트래픽 분석으로 연결을 설정하고 초기 블록을 다운로드 받고 블록 및 트랜잭션이 전파되는 과정을 검증하였다. 하지만 시간을 측정하거나 성능 분석은 실시하지 않았고 최근에 사용되는 CBR 동작을 분석을 하지 못했다.



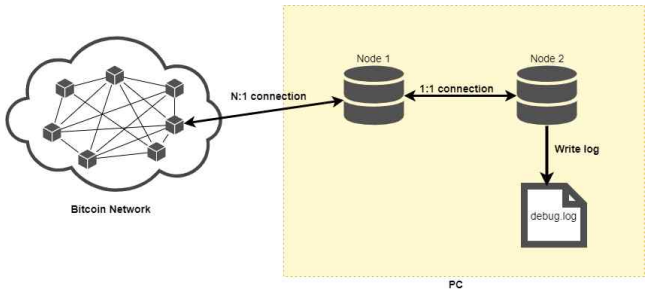
<그림 1> 비트코인 고 대역폭 압축 블록 전달

비트코인 네트워크에는 블록이나 트랜잭션 전파를 신속히 하기 위해서 특별히 고안된 릴레이 네트워크가 있다. 이 릴레이 네트워크의 대표적인 것이 FALCON, FIBRE이며 이 들은 비트코인 네트워크에 참여하는 노드이나 오직 전파 기능만 담당하고 전 세계에 걸쳐 고속 연결을 구축하여 전파 시간을 단축시킨다[12][13]. Kai 등은 이들 고속 전달 네트워크의 성능을 검증하였으며 77%의 성능 개선 효과가 있음을 밝혔다[14]. 본 연구는 고속 릴레이 네트워크와 연결에 대한 전달 성능은 검증하지 못하였고 향후 연구 과제 중의 하나이다.

Ⅲ. 실험 환경 및 실측 데이터

3-1. 실험 환경

실험 환경을 2개의 비트코인 노드로 구성하였다. <그림 2>은 구성된 네트워크를 보여주고 있다. Node 1은 비트코인 네트워크에 연결되어 있고 이 노드에 Node 2가 직접 연결되어 있다. Node 2에서 발생하는 로그는 분석을 위해서 저장된다.



<그림 2> 비트코인 블록 전파 측정을 위한 환경 구성

Node 1은 비트코인 네트워크에 연결되어 있으므로 비트코인 네트워크로부터 실제로 생성된 블록을 전달 받는다. Node 2는 오직 Node 1에만 연결되어 있으므로 Node 1이 받은 블록만 전달 받게 된다. 두 노드는 하나의 컴퓨터에서 도커 컨테이너(Docker Container)로 만들어졌다[15]. 이와 같이 구성한 이유는 2개 노드가 커널을 공유하는 도커로 동작함으로써

2개 노드의 시간이 동일하고 물리적인 네트워크 연결이 없으므로 네트워크 지연이 발생하지 않도록 하기 위해서 이다. Node 1과 Node 2는 네트워크 연결을 제외하고 모든 설정 및 동작은 폴 노드의 기본 값을 사용하였다.

노드가 실행되는 PC는 i5-7500T CPU, 8GB의 메모리를 사용하며 저장 공간은 SSD 1TB 하드디스크를 사용하고 OS는 우분투 18.04LTS를 사용하였다. 이 구성은 2개의 노드가 실행되기에 충분한 시스템 성능을 갖추고 있으며 다만 마이닝을 수행하지 않는다. Docker 컨테이너의 OS도 PC와 동일하게 우분투 18.04LTS를 사용하였으며 비트코인 코어는 "Bitcoin Core version v0.19.99.0-b9b58f8f6-dirty" 버전을 사용하였다[16]. 이 구성은 2020년 현 시점에서 가장 최근에 배포한 소프트웨어이다. 비트코인 코어를 수정하여 로그 값이 기록되는 시간을 기본 초 단위에서 마이크로 초 단위로 기록하도록 소스코드를 수정하였다. 비트코인 코어 프로그램을 실행 할 때 단순 로그 값 뿐만 아니라 네트워크 관련 즉 패킷 송수신 내용도 기록 하도록 옵션을 변경하였다. 두 노드는 최신 블록까지 동기화 된 이후에 전파되는 블록들을 기준으로 전달 시간을 측정하였다.

측정을 위하여 Node 2에서 만들어진 로그는 파일로 저장되었으며 로그는 문자열로 되어 있으며 메시지를 주고받은 로그의 형태는 다음과 같은 형식이다.

[TIME] [MESSAGE]

TIME = [YYYY'T'HH':'MM':'MM':'SS'Z'mmmmm]

MESSAGE = ["received"|"sending" MSG-TYPE (BBBB 'bytes') "peer"=X]

TIME은 밀리미터 초까지 표현하며 주고 받은 메시지는 수신의 경우 received로 송신의 경우 sending으로 표현되고 이후 MSG-TYPE으로 송수신한 메시지가 무엇인지 나타낸다. 추가로 교환한 메시지의 길이가 주어지고 송수한 상대 노드의 번호가 "peer=X"로 표현되는데 이 때 X의 값은 프로그램이 처음 실행된 후 연결한 노드의 번호로 로그의 앞 부분에서 어떤 노드인지 확인 할 수 있다.

Node 2에 저장된 로그의 예제는 <그림 3>과 같다. 이 예제에서 블록 높이가 618754번째를 전파하는 과정을 보여준다. 로그가 기록될 때 날짜와 시간이 기록되는데 기본 비트코인 코어에서는 초 단위가 가장 작은 단위로 기록하지만 마이크로초 단위까지 기록하도록 수정하였기 때문에 정밀하게 측정할 수 있다. 마지막에 peer 뒤에 작성되는 관련 노드 번호는 노드와 연결된 노드들을 번호를 붙여서 어떤 노드와 패킷을 교환하는지를 표시한다.

<그림 3> 비트코인 코어 수신 받는 노드에 기록된 로그

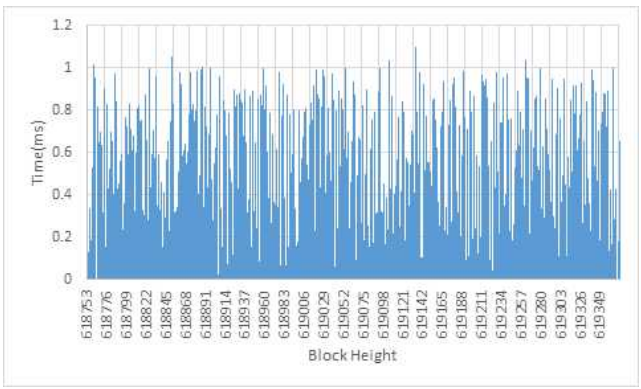
실험은 618753번 블록부터 619371번 블록까지 실시되었으며 생성된 로그의 양은 638MB이고 이 로그에는 메시지 교환에 대한 로그와 데이터 검증과 노드에서 발생하는 에러와 같은 노드 내부에서 동작하는 내용들도 포함되어 있다.

비트코인 코어 노드에서 기록 하는 로그 중 다른 노드와 데이터를 교환할 때 발생하는 메시지 중에서 동기화를 위한 트랜잭션과 블록에 관한 메시지가 존재한다. 트랜잭션에 관한 메시지는 트랜잭션 ID 값이 기록되어 있어 확인이 가능하며, 블록 동기화에 관한 메시지는 블록 전파 이후 노드에서 검증 과정을 하고 작성되는 메시지에 블록 정보가 기록되어 있어 확인이 가능하다. 하지만, 트랜잭션은 각 트랜잭션 메시지마다 ID 값이 기록되어 구별이 가능하지만 블록은 검증 이후 기록되는 메시지를 기준으로 이전에 기록된 블록 관련 메시지를 분석하였다. 블록 동기화가 끝난 시점부터 측정하여, 부족한 블록을 요청하는 과정을 보여주지 않고, 이후 새로 생성된 블록이 전파되는 과정이 진행된다. 동일한 패턴을 분석한 결과 cmpctblock, getblocktxn, blocktxn 순서로 로그에 기록되며, <그림 3>에서와 같이 cmpctblock 메시지로 시작한다. 이후에는 내부 동작 로그 Initialized PartiallyDownloadedBlock 메시지 로그가 기록되고 그 이후에 getblocktxn 메시지와 blocktxn 메시지가 기록된다. blocktxn 메시지가 기록되면 내부 동작 로그 Successfully reconstructed block 메시지 로그가 기록된 다음 추가 트랜잭션을 요청하고 블록 전파 과정이 이루어진다. 그리고 블록 정보에 관한 로그 메시지가 기록되며 전파된 블록 정보를 로그로 확인이 가능해진다.

노드에 기록되는 로그 중에서 3개의 블록 관련 메시지 로그와 2개의 노드 내부 동작 로그를 기준으로 블록 전파를 위한 과정으로 분석하여 블록 전파 시간을 로그 시간을 비교하여 측정 후 분석 하였다.

Ⅳ. 노드 간 전달 성능 분석

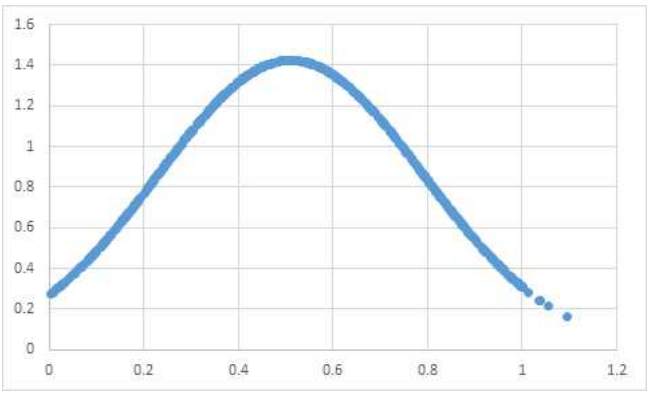
비트코인 동기화 과정에서 트랜잭션 전파 과정을 빼고 블록 전달을 위한 cmpctblock 패킷을 시작으로 blocktxn 패킷을 마지막으로 블록이 전달되는 시간을 측정했다. 블록 높이(Height)를 기준으로 618753번 블록부터 619371번 블록까지 총 618개의 블록에 대하여 측정하였으며 그래프로 표현한 것이 <그림 4>이다. 가로축은 블록 높이(height)를 세로축은 전달하는데 소요된 시간이다. 평균값은 0.5초 정도로 측정되고 최대 1초에서 최소 0.002초로 측정되었으며 표준편차는 0.28이다. 그림에서 육안으로도 전달시간의 변화가 매우 크게 보이고 이는 표준편차가 매우 큰 값으로 나온 이유이며 블록전달에 변화가 심하다고 판단되며 긴 전달 시간과 짧은 전달시간 간에는 어떤 이유로 이런 차이가 발생하였는지 확인이 필요하다.



<그림 4> 블록별 블록 전파 시간 측정 그래프

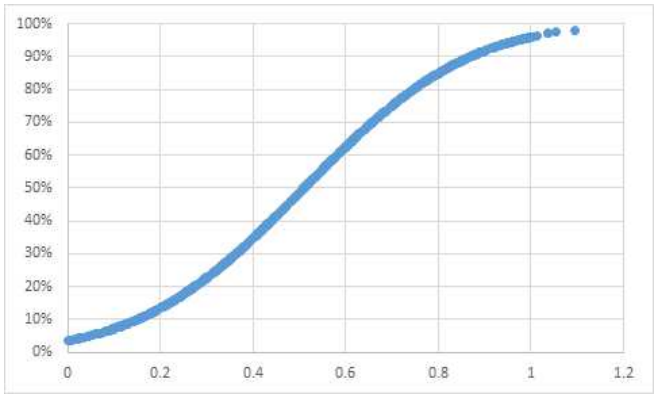
비트코인 블록 전파 시간을 측정해 그래프로 표현해 보았지만 분포도로 표현하여 블록 전파시간이 특정한 시간대에 밀집해 있는지 아니면 분포가 되어 있는지 확인하기 위해 정규분포와 누적정규분포(CDF)를 사용하여 그래프로 표현하여 블록 전파 시간이 어떻게 분포 되었는지 확인 할 수 있다.

우선 <그림5>는 측정한 블록 전파 시간을 발생 횟수로 표현한 그래프다. 가로축은 블록 전달 시간을 세로축은 전달하는데 정규 분포 도수 값이다. 그래프를 보면 평균 0.5초 기준으로 나뉘어 있는 모습을 볼 수 있고 모양이 한쪽으로 치우치거나 한쪽 꼬리 부분이 넓게 펼쳐지지 않아서 정규 분포로 볼 수 있다. 육안으로 보기에도 정규분포를 따르고 있으므로 정규성 검증은 불필요하다. 다만 블록별 전달 시간 그래프에서 확인한 바와 같이 그래프의 모양이 중심점에서 높지 않고 옆으로 많이 퍼져 있어 전달 시간 변화가 크다는 점을 알 수 있다. 대부분의 블록 전파 시간은 1초 이내의 시간이 소비되었고 그 이상의 시간은 적은 수의 결과 값을 보여준다. 즉, 비트코인 블록 전파에 있어서 거의 대부분 1초 이내의 시간을 소비하고 적은 수의 전달이 그 보다 큰 시간을 소비하기는 하지만 1.2초 이내에서 블록 전파가 이루어짐을 알 수 있다.



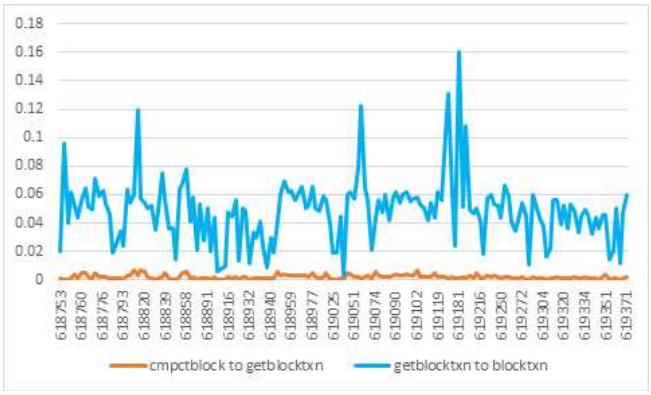
<그림 5> 노드 간 블록 전파 시간 정규분포 그래프

누적정규분포(CDF)로 블록 전파 시간 값을 그래프로 표현하면 <그림 6>와 같은 그래프가 그려진다. 가로축은 블록 전달 시간을 세로축은 누적정규 분포 도수 값이다. 평균시간 0.5초를 기준으로 분포되었던 정규분포 그래프와 유사하지만, 누적정규분포(CDF) 그래프를 보면 0.2초에서 0.8초 사이 대부분의 블록이 전파 되었고 나머지 20% 정도가 범위를 벗어나게 측정되기는 하지만 큰 시간차이를 보여주지는 않는다.



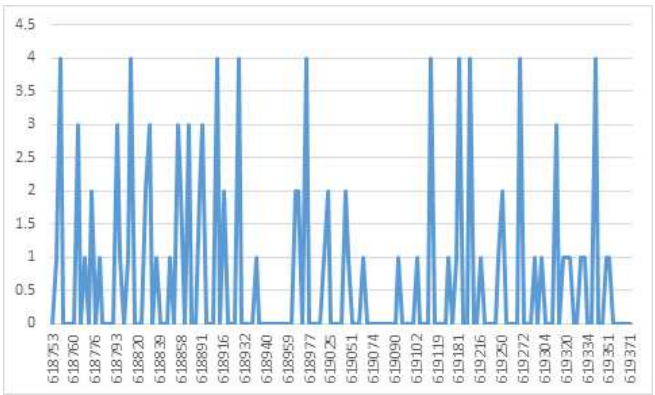
<그림 6> 노드 간 블록 전파 시간 누적정규분포(CDF) 그래프

<그림 1>을 보면 한 블록을 전달하는데 걸리는 시간은 2구간으로 나룰 수 있다. 첫째 구간은 cmpctblock 메시지와 getblocktxn 메시지 사이의 구간이고 다음 구간은 getblocktxn 메시지와 blocktxn 메시지 사이 구간이다. 첫째 구간은 블록에 포함된 트랜잭션을 확인해서 메모리 풀에 없는 트랜잭션을 요청하는데 소요되는 시간이며 둘째 구간은 이 트랜잭션들을 전달 받은데 소요되는 시간이다. 이 두 구간의 측정하여 각 그래프로 그린 것이 <그림 7>이다. 가로축은 블록 높이(height)를 세로축은 블록을 전달하는데 소요된 시간이다.



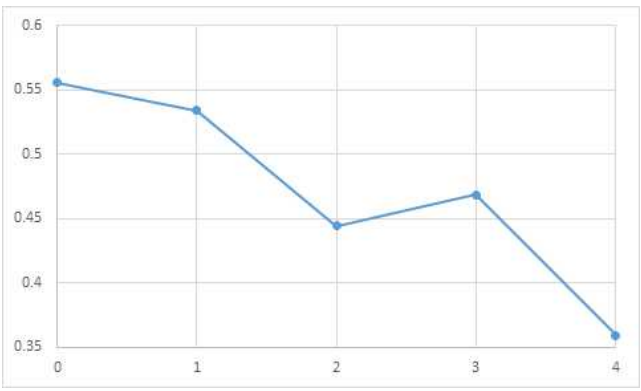
<그림 7> 블록 전파 메시지 구간 별 시간 차이

첫째 구간인 cmpctblock 메시지와 getblocktxn 메시지 사이 구간이 둘째 구간인 getblocktxn 메시지와 blocktxn 메시지 사이 구간이 더 적은 시간을 소요함을 볼 수 있다. 그리고 첫째 구간은 거의 일정하며 둘째 구간은 블록별로 상당히 많은 차이가 있음을 알 수 있다. 그리고 블록에 존재하지만 메모리 풀에 없어서 재요청을 한 트랜잭션의 수를 그래프로 그린 것이 <그림 8> 이다. 가로축은 블록 높이(height)를 세로축은 트랜잭션 재요청 수이다. 블록별 재요청 트랜잭션 수를 측정한 결과는 재요청의 횟수가 최대 4개까지 있으며, 평균 0.6개의 트랜잭션을 재요청하여 블록을 구성할 때 사용한다.



<그림 8> 블록별 재요청 트랜잭션 수

트랜잭션은 블록이 전파되기 전에 미리 전파 되어 메모리 풀에 저장되며 압축 블록을 전파 받았을 때 메모리 풀어 저장되지 않은 적은 수의 트랜잭션만 별도로 요청한다.



<그림 9> 재요청 트랜잭션 수와 블록 전달시간의 상관 그래프

블록별 재요청 횟수와 블록 전달 시간은 매우 밀접한 관계가 있을 것이

다. 그래서 블록별 재요청 트랜잭션 수와 블록 전달시간과의 상관 그래프가 <그림 9> 같다. 가로축은 재요청 트랜잭션 수이고 세로축은 해당 재요청 트랜잭션 수를 발생시킨 블록의 평균 전달시간이다.

그림에서 보는 바와 같이 매우 밀접한 관계를 보이며 Correl 함수를 이용하여 상관계수를 계산한 결과 -0.929234이다. 이는 매우 강력한 음(-)의 상관관계 이라는 뜻으로 블록 전파 시간은 트랜잭션 재전송 횟수가 많을수록 더 짧은 시간으로 측청 되는 것이다.

Ⅴ. 결론

비트코인 네트워크에서 블록 동기화를 위해 블록이 전파되는 과정을 패킷 단위로 분석하고 블록이 전파되는 시간을 측정하였다. 비트코인 네트워크에서 블록 전파는 cmpctblock 패킷을 시작으로 blocktxn 패킷을 마지막으로 이루어진다.

그리고 패킷을 기준으로 블록이 전파되는 시간을 분석하여 그래프로 표현하였다. 대부분의 블록은 1초 이내의 시간을 소비하여 전파를 진행하였고 대분의 전파 시간은 0.2초에서 0.8초 사이의 결과를 보여준다. 블록 전파 과정을 단계별로 나누어서 측정한 결과는 cmpctblock 메시지와 getblocktxn 메시지 구간보다 getblocktxn 메시지와 blocktxn 메시지의 구간이 더 많은 시간이 걸린다. 그리고 블록 전파 과정에서 재요청하는 트랜잭션의 수는 최대 4개로 측정되며 평균 0.6개의 트랜잭션을 재요청한다.

비트코인 노드를 단일 직접 연결로 구성하여 데이터를 측정하였지만, 향후 여러 노드가 연결된 상태에서 블록 전파 시간을 측정할 때 본 논문의 결과를 비교자료로 사용할 수 있다.

측정된 데이터는 시간에 따른 측정값으로 볼 수 있어 이를 기반으로 특정 경향(Trend), 주기(Cycle)가 있는지 그리고 불규칙성에 대한 분석을 위하여 향후 연구로 시계열 자료 분석을 실시할 예정이다. 이 분석을 발전시켜서 블록 전달 시간 간격과 블록 전달 시간과의 관계 등에 대한 분석을 통해 비트코인 네트워크에 상태나 비트코인 노드에서 발생하는 이상 행위 분석에 사용할 수 있다.

향후 연구로는 비트코인 노드에서 제공하는 로그 데이터를 이용한 방식이 아닌 패킷을 직접 수집하는 방식으로 단순히 설치된 노드 주변의 블록 전파 시간뿐만 아니라 원격 노드에 블록 전파 시간을 측정할 수 있도록 자동화 시스템을 구축하여 비트코인 전체 네트워크에서 블록이 전파되는 시간을 수집할 수 있도록 연구를 진행할 예정이다. 또한 직접 연결하여 분석한 방법을 네트워크상으로 서로 멀리 떨어진 노드에서 블록 전달시간을 측정하여 네트워크 연결이 블록 전달에 미치는 영향을 분석할 예정이다.

ACKNOWLEDGMENT

이 논문은 2020년도 정부(교육부)의 재원으로 한국연구재단의 지원과 (NRF-2018R1D1A1B07050380) 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539, 블록체인의 트랜잭션 모니터링 및 분석 기술개발)

참 고 문 헌

[1] Stoshi Nakamoto, “Bitcoin: A peer-to-peer electronic cash system”, 2008

[2] BENČIĆ, Federico Matteo; ŽARKO, Ivana Podnar. Distributed ledger technology: Blockchain compared to directed acyclic graph. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2018. p. 1569-1570.

[3] Block timestamp, https://en.bitcoin.it/wiki/Block_timestamp, 2019

[4] Auqib Hamid Lone, Roohie Naaz Mir, “Investigating and Analyzing Bitcoin Blockchain Protocol using Wireshark”, I. J. Computer Network and Information Security, 2018, 7, 36-43

[5] Ryunosuke Nagayama, Kazuyuki Shudo, and Ryohei Banno, “Simulation of the Bitcoin Network Considering Compact Block Relay and Internet Improvements”, arXiv:1912.05208, 2019

[6] ZHANG, Shijie; LEE, Jong-Hyouk. Double-spending with a Sybil Attack in the Bitcoin Decentralized Network. IEEE Transactions on Industrial Informatics, 2019.

[7] Soni, Aakanksha, and Saurabh Maheshwari. "A Survey of Attacks on the Bitcoin System." 2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS). IEEE, 2018.

[8] Muhammad Saad, Jeffrey Spaulding, Laurent Njilla, Charles Kamhoua, Sachin Shetty, DaeHun Nyang, and Aziz Mohaisen, "Exploring the Attack Surface of Blockchain: A Systematic Overview”, arXiv:1904.03487v1, Apr 2019

[9] CORALLO, Matt. BIP 152: Compact block relay. See https://github.com/bitcoin/bips/blob/master/bip-0152. mediawiki, 2016.

[10] Bitcoin Developer Reference, https://bitcoin.org/en/developer-reference, 2020

[11] Decker, Christian, and Roger Wattenhofer. "Information propagation in the bitcoin network." IEEE P2P 2013 Proceedings. IEEE, 2013.

[12] Otsuki, Kai, et al. "Effects of a Simple Relay Network on the Bitcoin Network." Proceedings of the Asian Internet Engineering Conference. 2019.

[13] Bi, Wei, Huawei Yang, and Maolin Zheng. "An accelerated method for message propagation in blockchain networks." arXiv preprint arXiv:1809.00455 (2018).

[14] Kai Otsuki, Yusuke Aoki, Ryohei Banno, Kazuyuki Shudo, “Effects of a Simple Relay Network on the Bitcoin Network”, Proceedings of the Asian Internet Engineering Conference, August, 2019, pp 41-49

[15] Docker, https://www.docker.com/, 2020

[16] Bitcoin core, https://github.com/bitcoin/bitcoin, 2020

스테이블 코인(stable coin)의 종류와 특징 분석

강창훈, 고경찬, 우종수, 홍원기
포항공과대학교

{chkang, kkc90, woojs, jwkhong}@postech.ac.kr

Analysis of Stable Coin Types and Characteristics

Changhoon Kang[1], Kyungchan Ko[1], Jongsoo Woo[2], James Won-Ki Hong[1]
[1]Department of Computer Science and Engineering, POSTECH
[2]Graduate School of Information Technology, POSTECH

요 약

비트코인이라는 첫 암호화폐가 등장한 이후로 많은 암호화폐들이 새롭게 생겨났다. 각각의 암호화폐들은 해당 프로젝트의 목적에 따라 다양한 특징과 동작 메커니즘을 가진다. 그 중 암호화폐의 가치가 일정하게 유지되는 것을 목표로 하는 암호화폐를 스테이블 코인 (stable coin)이라고 한다. 대부분의 암호화폐들은 가치의 변동성이 매우 큰 반면, 스테이블 코인은 가치의 안정성을 위한 메커니즘을 도입하여 큰 가치 변동없이 이를 일정하게 유지하도록 설계되었다. 특히나 최근 JP Morgan Coin 이나 페이스북의 Libra 등이 많이 탄생하면서 스테이블 코인의 중요도와 필요성이 더욱 높아졌다. 본 논문에서는 스테이블 코인들을 담보물의 종류에 따라 구분하여 설명하고, 각각을 비교, 분석한다. 또한 스테이블 코인이 유지할 목표 가격을 설정할 때, 기준으로 법정화폐 가격을 사용하고 있음으로써 생기는 문제점과 이를 해결할 수 있는 새로운 방안을 소개한다. 마지막으로 현재 스테이블 코인과 관련해 어떤 연구가 진행되고 있고, 앞으로 어떤 방향의 연구가 필요한지 논의한다.

Ⅰ. 서 론

비트코인 [1]은 가장 처음 등장한 암호화폐로 2009 년에 처음 발행되었다. 이의 뒤를 이어 많은 암호화폐들이 스마트 컨트랙트, 다양한 합의 알고리즘 등 새로운 특징들과 함께 등장하였다 [2]. 결과적으로 암호화폐들의 전체 시장 총액이 약 2,000 억 달러를 이미 넘어섰다 [3]. 암호화폐가 활성화 될수록 일반적인 거래에서도 암호화폐를 활용할 기회가 많아지지만, 암호화폐의 높은 가격 변동성 때문에 아직까지 잘 사용되지 않고 있다.

기본적으로 암호화폐들은 전자 지불 시스템이지만 이를 지불 방식으로 사용하는 사람은 거의 없다. 짧은 순간에도 급격하게 변동하는 암호화폐의 가치가 그 원인이다. 암호화폐로 물건을 구입하는 경우 거래 이후 발생할 수 있는 암호화폐의 가치 변동으로 인해 판매자와 구매자 모두가 잠재적인 손실이 발생할 수 있는 위험성을 가지게 된다. 이러한 이유로 아무도 굳이 기존 지불 시스템을 뒤로 한 채 가격 변동성이 큰 암호화폐를 거래에 사용하려고 하지 않는다.

이러한 단점을 해결하고 암호화폐를 전자 지불 수단으로 사용하기 위해 등장한 것이 스테이블 코인이다 [4]. 거래에 사용되기 위해서는 가치의 안정성이 있어야 한다는 바에 맞추어, 스테이블 코인은 가치 안정화를 위한 메커니즘을 통해 자신의 가치를 특정 가격으로 일정하게 유지되도록 한다. 따라서, 여러 원인들에 의해 스테이블 코인의 가치가 목표한 가격에서 벗어난다면 스스로 그가 가진 메커니즘을 통해 다시 회복한다. 스테이블 코인처럼 암호화폐의 가치 변동성 문제만 해결한다면, 이를 지불 수단으로 사용하였을 때 해외송금 처리 시간의 단축 및 낮은 수수료, 강력한 보안성 등 잠재적인 암호화폐의 특징들을 얻을 수 있다.

이런 장점들을 바탕으로 스테이블 코인은 탈중앙화 금융 사업 [5]에서 주요한 역할을 하고 있다. 해외로의 복잡한 송금 과정이나 까다로운 금융기관의 이용조건 등 기존 시스템의 문제점을 스테이블 코인을 활용한 탈중앙화 금융 사업을 통해 해결해 나가고 있다. 한 예로 페이스북은 Libra [6]라는 스테이블 코인을 발행하여, 기존 금융기관의 서비스를 이용하지 못하고 있는 전 세계 모든 이들에게 전자 지불 수단을 제공하려는 프로젝트를 진행중이다.

스테이블 코인은 특정 실물 자산이나 기존 화폐를 담보로 하여 발행하고, 어떤 것을 담보로 하는지에 따라 구분할 수 있다. 이와 달리 아무 것도 담보로 두지 않는 무담보형 스테이블 코인 역시 존

재한다.

본 논문에서는 스테이블 코인의 종류와 특징에 대해 설명하고 이들을 비교 및 분석한다. 또한 스테이블 코인이 현재의 가치를 지속적으로 보장하기 위해 필요한 점에 대해 논의하고, 기존에 진행되고 있거나 앞으로 필요한 연구들에 대해 이야기한다.

Ⅱ. 스테이블 코인의 분류

스테이블 코인은 발행할 때 무엇을 담보로 두었는지에 따라 구분할 수 있다. 각 분류마다 담보물의 특성에 의해 스테이블 코인이 가지는 특징 역시 달라진다. 아래에서는 대표적인 네 가지 방식에 대해 설명하고 각 방식이 가지는 특징에 대해 소개한다 [4].

1. 법정화폐 담보 스테이블 코인

특정 기관이나 회사가 법정화폐(달러, 유로 등)를 예치해두고 그 가치만큼의 스테이블 코인을 발행하여 유통시키는 형식이다. 가장 대표적인 방법이며 언제든 법정화폐와 1:1 환전이 가능하다. 대표적인 예시로 Tether(USDT) [7]는 1 USDT 의 가치를 1 US 달러와 동일하게 유지시키면서 항상 해당 가격에 환전을 해주고 있다. 다만 이를 발행한 특정 기관이 예치금 및 운영과 관련한 모든 규칙들을 관리하는 등 시스템이 중앙화되어 있기 때문에 언제든 자신들이 원하는 방향으로 운영 방침을 변경할 수 있다는 위험성이 존재한다.

2. 실물자산 담보 스테이블 코인

위에서 설명한 형식에서 법정화폐 대신 금이나 은과 같은 실물자산을 예치해두고 그 가치만큼의 스테이블 코인을 발행해 유통하는 방식이다. 따라서 이 역시 발행 기관에 시스템이 중앙화 되어 있다는 문제점이 있다. 한 예시로 Digix Gold Token (DGX) [8]은 예치해둔 금 1 그램 당 1 DGX 를 발행하고 있다.

3. 암호화폐 담보 스테이블 코인

스테이블 코인의 수요자가 직접 담보로 암호화폐를 예치해두고 스테이블 코인을 대출받는 형태이다 [9]. 스테이블 코인을 발행 받은 수요자는 대출 기한 내에 다시 같은 양의 스테이블 코인을 갚아야 한다. 이 모든 과정은 스마트 컨트랙트를 통해 진행되어, 운영 기관이 이미 담보를 받고 발행해준 코인에 대해 운영 규칙을 변경할 수 없다. 따라서 다른 스테이블 코인들이 가지고 있던 중앙화의 문제를 해결하였다. 하지만 법정화폐, 실물자산과는 다르게 암호화폐는 가격의 변동성이 매우 크기 때문에 담보로 받아둔 암호화폐의 가격이 하락하여 스테이블 코인을 발행 받은 수요자의 상환 의지를 없앨 가능성이 있다. 이런 경우를 대비해 발행 기관의 의지만으로도 강제로 상환이 가능하도록 하는 기능을 갖게 하거나, 처음 발행 시에 담보로 받은 암호화폐의 가치보다 조금 낮은 가치만큼의 스테이블 코인을 발행해 빌려줌으로써 수요자가 대출을 상환할 동기를 갖도록 하는 경우도 있다. 예를 들어 MakerDAO (DAI) [10]는 최소담보비율을 설정하여 담보로 받은 암호화폐의 총 가치에 설정된 비율을 적용하여 담보보다 낮은 가치의 스테이블 코인을 발행해준다.

4. 무담보 스테이블 코인

앞서 언급된 다른 스테이블 코인들과는 다르게 아무런 담보물 없이 발행된다. 스테이블 코인이 유지할 목표 가격을 설정하고, 시스템이 수요량 변동에 따라 자동적으로 공급량을 조절하도록 해 목표 가격을 유지할 수 있도록 하였고, 이로써 중앙화를 방지하였다. 수요량에 따라서 스테이블 코인을 제한없이 발행할 수 있기 때문에 쉽게 가격 변동성을 낮출 수 있다. 한 가지 큰 단점으로는 스테이블 코인의 미래 성장에 대한 신뢰를 잃게 되는 순간 해당 스테이블 코인은 다시 회복이 불가능한 시스템이 되어버린다. Basis (Basecoin) [11]의 예시를 살펴보면, 가격이 목표 가격보다 낮아졌을 때 이를 회복하는 메커니즘은 사람들로부터 코인을 회수하면서 그것보다 높은 가치의 채권을 발행함으로써 유통량을 줄이는 것인데, 미래에 대한 전망이 어두운 경우 채권을 구매하려는 의지가 없어지기 때문에 해당 메커니즘을 통한 목표 가격으로의 회복이 불가능해진다.

Ⅲ. 스테이블 코인 비교 및 분석

앞에서 제시한 네 가지 분류의 스테이블 코인을 담보물의 안정성과 시스템의 탈중앙화 여부 측면에서 비교 및 분석할 수 있다. 아래의 표 1 은 이를 간단히 정리한 것이다.

표 1. 담보물에 따른 스테이블 코인의 특성

담보물	안정성	탈중앙화
법정화폐	O	X
실물자산	O	X
암호화폐	X	O
무담보	O	O

1. 담보물의 안정성

법정화폐와 실물자산은 가치의 큰 변동이 없는 안정적인 담보물이다. 해당 가치와 동일한 양만큼의 스테이블 코인만을 발행하기 때문에 코인의 소유자가 원할 때면 언제든 그 가치만큼의 법정화폐 또는 실물자산으로의 환전이 가능하다. 거래소 상에서 코인의 가격이 조금 변동하더라도 그와 관계없이 원래의 가치를 보장받을 수 있기 때문에 해당 스테이블 코인 역시 가치의 안정성을 얻을 수 있다.

무담보의 스테이블 코인 역시 위와 같은 경우라고 분류할 수 있다. 안정된 자산을 담보로 두는 것은 아니지만, 유동적인 공급량 조절을 통해 코인의 가치가 일정하게 유지되도록 하기 때문이다. 따

라서 언제든 코인의 가치를 동일하게 인정받을 수 있어 안정성을 얻을 수 있다.

암호화폐들은 대부분 안정성이 낮고 가치의 변동이 크다. 따라서 담보물로 사용되었을 때 스테이블 코인의 안정성에 긍정적인 영향을 주지 못하고, 이를 예치해둔 상태로 동일 가치의 스테이블 코인을 발행하는 방법은 매우 부적절하다. 그 결과 다른 종류의 스테이블 코인과는 달리 대출 형태의 시스템이 사용된다. 하지만 이 형태 역시 암호화폐 가치의 불안정성으로 인해, 대출 기한이 모두 지나지 않더라도 강제 상환이 이루어질 수 있다거나 담보로 둔 암호화폐의 가치보다 낮게 스테이블 코인을 발행해주는 등의 제한을 두고 있다.

결론적으로 다른 담보물들과 비교했을 때, 암호화폐는 안정성 제공 측면에서 스테이블 코인 발행을 위한 담보물로 사용하기에 그다지 좋지 못한 선택이라고 할 수 있다.

2. 시스템의 탈중앙화

법정화폐와 실물자산은 스테이블 코인을 발행하는 특정 기관이 예치금, 운영 정책 등을 모두 관리하고 마음대로 다룰 수 있기 때문에 시스템이 중앙화 되어있다고 볼 수 있다.

이와는 반대로 암호화폐를 담보로 하는 스테이블 코인의 경우 모든 과정이 스마트 컨트랙트를 통해 이루어지기 때문에, 발행 기관이 담보물을 빼돌리거나 이미 발행해준 코인의 환전에 대한 운영 정책을 수정할 수 없어 탈중앙화 되어있다고 볼 수 있다.

담보물을 두지 않는 스테이블 코인 역시 발행 기관은 예치된 자산을 아무것도 소유하고 있지 않고, 탈중앙화 되어있다고 볼 수 있다. 운영정책을 변경할 수는 있지만 이를 통해서 발행 기관이 얻을 수 있는 이득은 거의 없으며, 오히려 스테이블 코인에 대한 신뢰도를 하락시켜 시스템이 붕괴되는 역효과를 가져올 수도 있어 기존 자동화된 시스템이 잘 동작하는 것에만 집중하도록 만들 수 있다.

결국 앞서 설명한 중앙화 된 시스템을 가진 두 스테이블 코인은 특정 기관에 의해 가치가 완전히 없어질 수도 있는 위험성을 안고 있다.

Ⅳ. 스테이블 코인의 가치 보존

현재 대부분의 스테이블 코인들은 법정화폐인 US 달러를 기준으로 하여 목표 가격을 설정하고 있다. 동일한 양의 화폐로 구매할 수 있는 재화의 크기를 구매력 [12]이라고 하는데, 물가가 상승할수록 당연히 화폐의 구매력은 떨어진다. 따라서 대부분의 법정화폐들이 가진 구매력은 과거부터 지속적으로 점점 떨어져왔다. 즉 1 US 달러로 현재 구매할 수 있는 재화의 양이 과거에 비해 감소했듯이, 미래에도 현재보다 줄어들 것이 분명하다. 이때 스테이블 코인의 가격을 지금처럼 법정화폐 가치에 맞추어 발행하는 경우, 현재 보유한 스테이블 코인의 구매력 역시 미래에 감소한다는 문제가 발생한다. 스테이블 코인을 자산을 저장하는 하나의 수단으로 바라보았을 때 저장된 자산의 가치가 점점 하락하는 것과 동일하다.

이에 대한 해결책으로 소비자 물가를 기준 삼아 목표 가격을 설정하는 방법이 있다. 이때 소비자 물가를 잘 나타내는 소비자 물가 지수 (CPI, Consumer Price Index) [13]라는 지표를 기준으로 하여 목표 가격을 설정한다. 이 경우 물가 상승에 따른 구매력의 변화가 발생하지 않기 때문에 스테이블 코인의 가치가 미래에도 그래도 유지될 수 있다. 실제로 진행되었던 Basis [11] 와 지금도 진행중인 Xank [14] 라는 스테이블 코인 프로젝트는 백서를 통해 현재는 아직 아니지만 프로젝트의 다음 단계에서 소비자 물가 지수를 스테이블 코인 가격 설정의 기준으로 사용할 것이라 밝혔다.

스테이블 코인의 사용을 활성화하고 먼 미래까지 지속적인 사용이 가능하도록 하려면 위와 같은 해결책의 도입이 필요하다.

Ⅴ. 스테이블 코인 연구

현재까지는 기존 스테이블 코인들의 특징과 동작 원리에 대해 조사 및 분석하거나, 새로운 방식의 스테이블 코인 모델을 제안하는 연구가 진행되어 왔다. 전자의 경우 대체로 스테이블 코인의 개념에 대해 소개하거나 특정 스테이블 코인을 분석하여 정리한 논문들이 많았다. 후자는 스테이블 코인의 새로운 가치 안정화 메커니즘이나 전체적인 구조 등을 제안해 개선된 스테이블 코인 시스템을 보이고 있다. 하지만 이러한 연구는 논문들보다 새로운 스테이블 코인 프로젝트들로부터 더욱 활발히 이루어지고 있다.

Mita [4]는 스테이블 코인의 개념 및 종류를 설명하고, 대표적인 예시들을 들어 각각의 가치 안정화를 위한 메커니즘에 대해 자세히 소개한다. 경제학적 이론들을 토대로 스테이블 코인의 가격을 안정화시킬 수 있는 방법들을 설명하고, 각 방법을 사용한 예시들을 제시한다.

Moin [15]은 스테이블 코인을 디자인하는 방법론을 제시한다. 스테이블 코인을 디자인하는 것에 있어 고려해야 하는 구성 요소들을 선정하고, 기존 스테이블 코인들을 해당 요소들로 나누어 분석하였다. 이를 통해 각각의 장단점들에 대해 이야기하고 앞으로 스테이블 코인이 나가야 할 방향에 대해 논의하고 있다.

Ⅵ. 결론

본 논문에서는 스테이블 코인을 담보물의 종류에 따라 구분하고 각각을 비교, 분석하였다. 또한 스테이블 코인의 목표 가격 설정에 있어서 소비자 물가 지수를 기준 지표로 사용하는 방법에 대해 소개하였다.

스테이블 코인은 그 중요성이 대두되면서 최근 앞서 언급한 프로젝트들이 새롭게 생겨나고 있다. 하지만 아직 이에 대한 학문적인 연구는 부족하고, 대부분 진행되고 있는 프로젝트들에 대한 분석이

주를 이루고 있다. 스테이블 코인의 발전을 위해서는 가치 안정화를 위한 새로운 메커니즘, 스테이블 코인의 새로운 활용방안 등에 대한 연구가 여전히 필요하다고 판단된다. 앞으로 본 논문에서 언급한 가격 안정성, 탈중앙화, 가치 보존을 고려한 목표 가격 설정 등의 요소들을 설계 과정에서 모두 고려하여야, 먼 미래까지 지속 가능한 스테이블 코인이 등장할 수 있을 것이다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 임 (No.2018-0-00539)

참 고 문 헌

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).

[2] Sockin, Michael, and Wei Xiong. A model of cryptocurrencies. No. w26816. National Bureau of Economic Research (2020).

[3] 2013 coinmarketcap.com. See https://coinmarketcap.com/ (accessed 7 April 2020).

[4] M. Mita, K. Ito, S. Ohsawa, H. Tanaka, "What is Stablecoin?: A Survey on Price Stabilization Mechanisms for Decentralized Payment Systems." arXiv preprint arXiv:1906.06037 (2019).

[5] Chen, Yan, and Cristiano Bellavitis. "Blockchain disruption and decentralized finance: The rise of decentralized business models." Journal of Business Venturing Insights 13 (2020): e00151.

[6] Libra White Paper (Revised 21 Jan. 2020). Available at https://libra.org/en-US/white-paper/

[7] Tether: Fiat currencies on the Bitcoin blockchain, *Tether white paper* (2016). Available at https://tether.to/wp-content/uploads/2016/06/TetherWhitePaper.pdf

[8] Eufemio, Anthony C., Kai C. Chng, and Shaun Djie. "Digix's whitepaper: The gold standard in crypto assets." (2018).

[9] Kondova, Galia, Christian Bolliger, and Erich Thammavongsa. "Stablecoins: Types and Applications." Available at SSRN 3553296 (2020).

[10] Team, Maker. "The Dai Stablecoin System." (2017). Available at https://makerdao.com/en/whitepaper/

[11] Basis: A Price-Stable Cryptocurrency with an Algorithmic Central Bank (2018). Available at https://www.basis.io/basis_whitepaper_en.pdf

[12] Fisher, I., & Brown, H. (1911). The Purchasing Power of Money. Yale University.

[13] Boskin, Michael J., et al. "Consumer prices, the consumer price index, and the cost of living." Journal of economic perspectives 12.1 (1998): 3-26.

[14] K. Ryu Hyun, C. Billy, L. Ryan, and K, Sung "Xank: A Treasury-backed Stability-guaranteed Cryptocurrency" (2018). Available at https://xank.io/

[15] Moin, Amani, Emin Gün Sirer, and Kevin Sekniqi. "A Classification Framework for Stablecoin Designs." arXiv preprint arXiv:1910.10098 (2019).

체인코드 메세지 배치 처리를 통한 하이퍼레저 패브릭 성능개선에 관한 연구

이정원, 박세진
계명대학교 컴퓨터공학과

ljw6828@gmail.com, baksejin@kmu.ac.kr

A Study On Performance Improvement Of Hyperledger Fabric Through Batched Chaincode Message

JungWon Lee, Sejin Park
Department of Computer Engineering, Keimyung Univ.

ljw6828@gmail.com, baksejin@kmu.ac.kr

요 약

본 논문은 프라이빗 블록 체인 오픈소스 프로젝트 중에 하나인 하이퍼레저 패브릭 기술의 성능개선을 목표로 하고 있다. 최근 중요해지고 있는 데이터 관리의 탈중앙화를 블록체인으로 실현하고 있다. 블록체인은 데이터 증명, 보관, 은행 등 여러 분야에 사용되고 있는 기술인데 항상 성능 이슈가 언급된다. 거래에 대해 빠른 속도로 합의하여 원장에 기록하여야 은행, 카드 사용과 같은 초당 수 만 트랜잭션을 발생시키는 시스템에서 원활하게 사용할 수 있기 때문이다. 하이퍼레저 패브릭은 클라이언트가 생성한 트랜잭션을 네트워크의 Peer 노드에 전송하고 이전 기록과 함께 해시 처리하여 분산 원장에 데이터를 기록하고 블록으로 만들어 체인의 형태로 저장한다. Peer 노드는 여러 클라이언트에게 받은 트랜잭션들을 모으고 검증을 위한 체인 코드를 실행하게 되는데 검증이 완료된 트랜잭션들은 Orderer 노드로 전송되어 블록으로 만들어진다. 이때 만들어진 블록은 여러 Peer 노드들에 해시 값을 비교하여 블록의 검증작업을 거쳐 데이터의 무결성을 확인한다. 클라이언트가 생성한 트랜잭션이 검증을 위해 전달되는 과정에서 체인 코드 메세지를 주고 받는 과정을 배치 처리하여 네트워크 지연 시간과 전송 횟수를 줄여 하이퍼레저 패브릭의 성능을 개선하는 연구를 진행하였다. 성능의 결과는 초당 처리되는 트랜잭션 수(Transactions per second, TPS)로 나타내는데 실험 결과 Send Rate(TPS)가 2302→4244 로 약 2 배 가까이 대폭 증가 하였고, Throughput(TPS)는 1677→2244 로 약 1.5 배정도 크게 증가했다. 실험 결과 Send Rate, Throughput 가 큰 폭으로 증가했기 때문에 체인코드 메세지를 배치 처리가 훨씬 높은 성능을 낼 수 있다.

Ⅰ. 서 론

본 논문에서는 프라이빗 블록 체인 오픈소스 중 하이퍼레저 패브릭[1]의 성능 향상에 대해 연구하였다.

블록 체인 기술은 데이터를 '블록'이라고 하는 단위로 묶어 P2P 방식을 기반으로 생성된 체인 형태의 연결고리를 이용하여 묶어 분산 저장하여 누구라도 임의로 수정할 수 없고 누구나 변경의 결과를 열람할 수 있는 분산 컴퓨팅 기술 기반의 원장 관리 기술이다.

블록 체인 기술에는 퍼블릭과 프라이빗으로 구분된다. 퍼블릭 블록 체인은 최근 비트코인, 이더리움 등에 사용되는 시스템으로 누구나 네트워크에 참여하여 원장을 사용하여 거래 할 수 있는 형태를 말한다. 프라이빗 블록 체인은 네트워크 안에서 자격 증명을 한 멤버들만 참여할 수 있는 블록 체인 네트워크 형태를 말하고 네트워크에 참여한 멤버들의 서로간의 합의가 있어야만 발생된 거래를 증명하여 거래를 완료 할 수 있다. 이는 임의의 사용자가 악의적인 거래를 발생시켰다 하더라도 데이터의 무결성을 합의할 수 있는 다른 멤버들 덕분에 데이터들이 수정이나 삭제, 손상 등이 없었다고 증명 할 수 있게 되어 데이터를 보호 할 수 있다.

블록 체인 기술은 가상 화폐에 많이 사용되는데 퍼블릭 블록 체인들 중 비트코인[2]의 경우 7tps, 이더리움[3]의 경우 20tps 정도 밖에 처리 할 수가 없다. 반면 비자 시스템의 경우 초당 약 24,000 트랜잭션을 처리할 수 있다. 하이퍼레저 패브릭은 비자 시스템에 비해 느리고 비트코인, 이더리움 보다는 빠른 약 450tps 의 성능을 가지고 있다. 은행과 같은 상용 서비스에 사용되기에는 아직 성능이 부족하기 때문에 본 논문에서는 체인

코드[4] 메세지 배치 처리를 통해 하이퍼레저 패브릭의 성능 개선을 목표로 하고 있다.

하이퍼레저 패브릭은 프라이빗 블록 체인 기술로써 자격 증명을 거친 멤버들로만 이루어질 수 있고 기술적으로 비트코인과 이더리움보다는 빠른 TPS 를 가지고 있지만 비자 시스템과 같은 속도는 도달하지 못하였다. 하이퍼레저 패브릭의 프로세스는 클라이언트가 발생하는 트랜잭션을 Peer 노드가 체인 코드를 통해 검증하고 검증된 트랜잭션을 Orderer 노드로 보내 트랜잭션을 순서대로 블록으로 묶어 처리 하는 프로세스 흐름을 가지고 있다. 이때 트랜잭션이 발생할 때마다 검증을 위해 네트워크를 통해 체인 코드 메세지를 보내게 되는데 본 논문에서 제안하는 체인 코드 메세지 배치 처리를 통해 성능을 개선할 수 있는 아이디어를 제안한다.

Ⅱ. 디자인

Ⅱ-Ⅰ. 배경지식

하이퍼레저는 리눅스 재단의 주도 하에 산업간 블록 체인 기술 발전을 위해 조직된 전세계적인 오픈소스 협업 활동 중 하나이다. 그 중에서 하이퍼레저 패브릭은 모듈형 아키텍처 기반의 블록 체인 애플리케이션 또는 솔루션 개발을 위해 설계된 비즈니스 블록 체인 프레임워크로서 합의 프로토콜, 멤버십 서비스 등 구성요소의 플러그 앤 플레이를 지원하는 프라이빗 블록 체인 오픈소스 플랫폼이다. 프라이빗 블록 체인은 멤버십 서비스를 통하여 허가형 네트워크를 구축할 수 있고 멤버십 당사자들에게만 거래가 공개되어 기밀을 유지할 수 있다. 하이퍼레저 패브릭은 네트워크에 참여한 모든 참여자에게 트랜잭션에 대한 원장을 단 한 번만 기록, 분산 저장한다. 또한 원장에 한 번 기록되면 어떤 참가자도 트랜잭션을 변경하거나 위조할 수 없다. 트랜잭션 처리를 하기 위해서는 스마트 컨트랙트를 실행하여 트랜잭션에 대한 조건, 실행 로직 등을 정의할 수 있는데 하이퍼레저 패브릭에서는 체인 코드를 통해 구현, 실행한다.

하이퍼레저 패브릭의 네트워크에는 크게 Peer, Orderer 로 구성된다. 트랜잭션의 기록은 블록에 저장하며 최신 상태는 데이터베이스에 저장하고 트랜잭션은 Ordering 서비스와 합의한다. 이때 트랜잭션이 Orderer 에게 제안되기 이전에 Endorse 되어야 한다. Orderer 들은 Ordering Service 라는 통신 서비스를 제공하고 Peer 들과 Application 들에게서 트랜잭션을 받고 블록을 전달한다. Peer 들은 원장 상태와 체인 코드를 관리하는 네트워크 노드로써 Endorser, Committer, Submitter 의 역할을 할 수 있다. Endorser 는 항상 Committer 가 되고 Endorser 는 트랜잭션을 수행하고 승인하고 Committer 는 Endorsements 를 검증하고 트랜잭션 결과의 유효성을 체크한다.

하이퍼레저 패브릭을 활용한 애플리케이션 흐름은 클라이언트가 Invoke/Query 트랜잭션을 발생시키면 Peer 노드에서 체인 코드 실행을 통해 스마트 컨트랙트가 실행되고 검증된 트랜잭션들이 Ordering 을 위하여 네트워크 내의 Orderer 에게 전달된다. 이때 Orderer 는 전달받은 여러 트랜잭션을 순서대로 블록을 만들어 다시 Peer 들에게 전달한다. 전달 받은 블록들은 체인의 형태로 연결되어 분산 저장된다.

하이퍼레저 패브릭의 데이터 통신은 HTTP/2 을 활용한 gRPC[5]를 이용한다. gRPC 는 RPC(Remote Procedure Call)의 빠른 장점을 가진 원격 프로세스 호출 프레임워크이다. 원격의 프로시저를 클라이언트에서 호출할 수 있도록 해준다. gRPC 를 이용하여 노드들의 도커[6] 컨테이너 내부에서 원격 프로시저를 호출하는 방식으로 프로세스를 실행한다.

Ⅱ-Ⅱ. 하이퍼레저 패브릭 체인코드 메세지 배치처리를 통한 제안

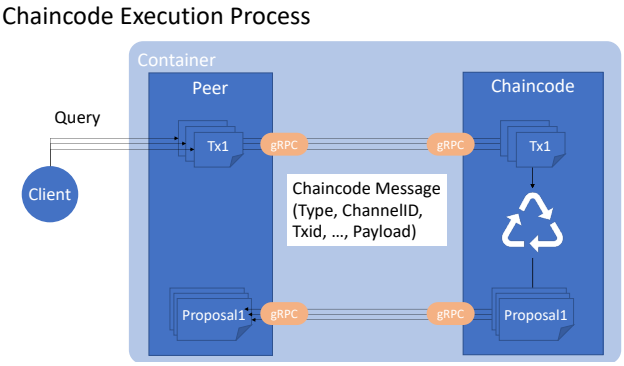


Figure 1 기존 체인 코드 프로세스 흐름도

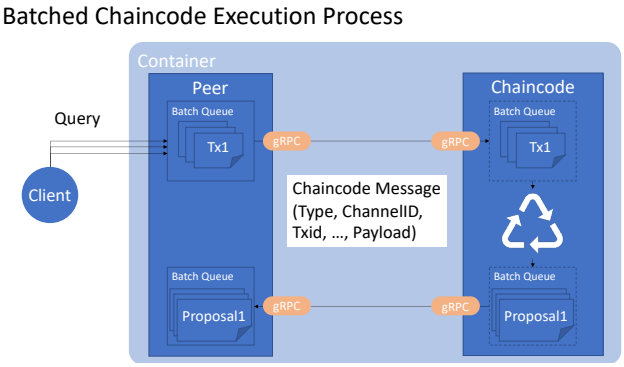


Figure 2 배치 처리 된 체인 코드 프로세스 흐름도

하이퍼레저 패브릭은 클라이언트가 발생시킨 트랜잭션을 Peer 노드에서 체인 코드를 통해 검증한다. Peer 노드에서 체인 코드를 실행하기 위해 gRPC 를 통해 체인 코드 실행에 필요한 데이터 셋을 주고 받는다. 이때 트랜잭션 당 체인 코드 메세지가 발생되고 gRPC 를 통해 네트워크에

전송된다. 이와 같은 기존의 체인 코드 실행 방식은 매번 새로운 통신을 연결 하여 체인 코드 메세지를 주고 받게 된다. 체인 코드 실행 프로세스를 배치 처리하여 여러 트랜잭션에서 발생한 다수의 데이터 셋을 1 번에 전송하고 처리하여 성능 향상을 기대할 수 있는 아이디어를 제안한다.

배치 처리란 순서대로 실행되는 트랜잭션을 하나씩 처리하는게 아니라 트랜잭션을 하나의 데이터 셋으로 묶어 일괄적으로 전송/처리하는 방식을 말한다.

Peer 노드에서 체인 코드 메세지를 1 개의 데이터 셋으로 묶어 저장하고 일정 시간 또는 배치 사이즈에 가득 차게 되면 체인 코드를 실행하기 위해 gRPC 통신으로 배치 처리된 데이터를 1 번만 전송하게 된다.

체인 코드 프로세스에서 배치 처리된 데이터를 받아 각각의 트랜잭션들로 풀어낸 후 각 트랜잭션을 체인 코드를 이용해 검증한다. 검증된 트랜잭션과 결과들은 다시 배치 처리 하여 gRPC 를 통해 리턴 된다.

Ⅲ. 실험

Ⅲ-Ⅰ. 실험 환경

실험 환경은 Table1 에 명시되어 있다. 본 논문에서는 하이퍼레저 패브릭의 성능 개선 실험을 위해 필요한 노드들을 도커를 통해 가상화하여 실행하였다. 네트워크 환경은 각 도커 노드들이 같은 로컬 서버 내에서 실행되고 있기 때문에 체인 코드 메세지를 전송할 때에 외부 통신 없이 동일한 서버 내 gRPC 통신으로 실험이 이루어져 도커 네트워크 성능과 gRPC 네트워크 성능의 영향만 받았다. 하이퍼레저 패브릭의 릴리즈 버전은 v1.1 이고 성능 측정은 같은 하이퍼레저 프로젝트들 중 하이퍼레저 캘리퍼[7]라는 하이퍼레저 패브릭 성능 측정 도구 오픈소스를 사용하였다.

Table 1 Experiment Environment

H/W	CPU	Intel(R) Xeon(R) CPU E5-2697 v3 @ 2.60GHz
S/W	OS	Ubuntu 16.04.6 LTS
	Docker Version	19.03.8
	Hyperledger Fabric Version	v1.1
	성능 측정 방법	Hyperledger Caliper

Ⅲ-Ⅱ. 실험 결과

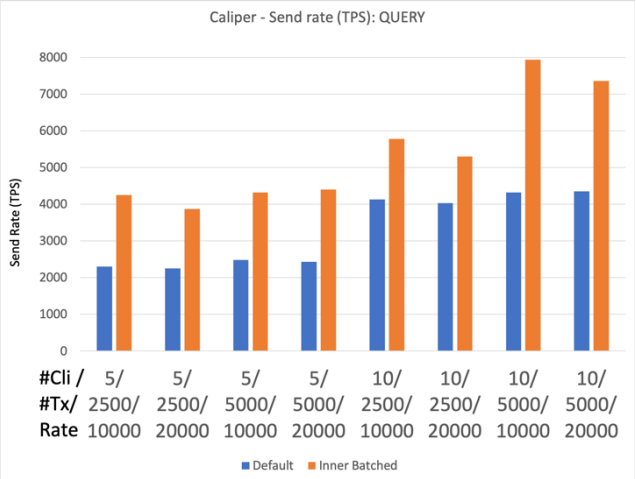


Figure 3 Send Rate(TPS) Query 측정 결과

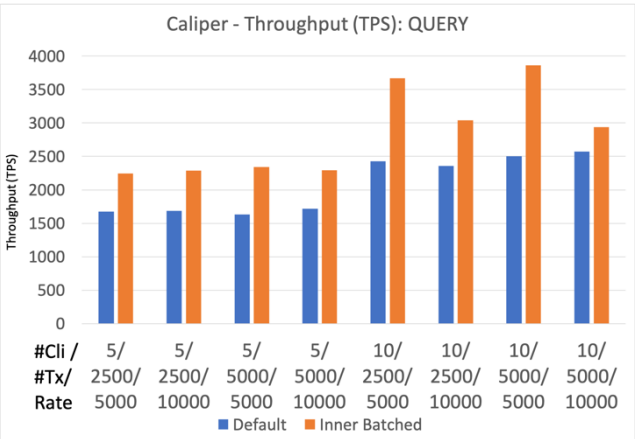


Figure 4 Throughput(TPS) Qurey 측정 결과

Figure3,4 는 하이퍼레저 캘리퍼를 통해 성능을 측정하였고 하이퍼레저 패브릭 네트워크의 채널 1 개일 때 클라이언트, 트랜잭션, Rate 를 다르게 하며 얻은 실험 결과이다. OPEN/QUERY 명령어를 통해 각 실험 상황에서의 결과를 확인했다.

Figure3 를 보면 클라이언트 수는 5 개, 트랜잭션 수는 2500 개, Rate 가 10000 일 때 Send Rate(TPS) 가 2302→4244 로 약 2 배 가까이 대폭 증가했다. 이와 같이 클라이언트가 5 개일 때는 약 2 배정도의 성능 향상이 있었고 클라이언트가 10 개일 때는 약 1.5 배의 성능 향상을 보여주었다. 클라이언트 개수가 증가했을 때 배치 처리의 효과가 더욱 커지는 것을 볼 수 있다. 또한 같은 클라이언트 개수일 때 트랜잭션과 Rate 가 증가하면 Tx 증가 폭이 월등히 높아진다.

모든 실험에서 Default Case 보다 Batched Case 가 훨씬 좋은 성능을 보여주고 있다.

Figure4 는 클라이언트가 5 개, 트랜잭션 수가 2500 개, Rate 가 5000 일 때 Throughput(TPS)이 1677→2244 로 약 1.5 배 에 달하는 성능 향상을 보여주고 있다. 클라이언트 수가 10 개로 증가한 상황에서도 500 TPS 이상의 증가 폭을 보여주면서 최대 약 1.5 배 정도의 성능 향상을 보여주고 있다. Throughput 또한 클라이언트 개수가 증가했을 때 배치 처리의 효과가 더욱 커지는 것을 볼 수 있다. 또한 Send Rate 와 같이 클라이언트 개수가 같을 때 트랜잭션과 Rate 가 증가하면 Tx 증가 폭이 월등히 높아진다. Throughput(TPS)도 모든 실험에서 Default Case 보다 Batched Case 가 훨씬 좋은 성능을 보여주고 있다.

IV. 결론 및 향후 연구 방향

본 논문에서는 하이퍼레저 패브릭의 체인 코드 메세지 배치 처리를 통해 트랜잭션을 전송하는 횟수는 줄이고 한번에 더 많은 트랜잭션을 처리할 수 있게 하는 성능 개선 연구를 진행했다. 기존 프로세스에서는 매 트랜잭션마다 네트워크 통신이 이루어져 여러 트랜잭션을 한번에 처리할 수 없었는데 체인코드 메세지 배치 처리를 통해 여러 트랜잭션을 한번에 처리할 수 있게 되어 같은 하이퍼레저 패브릭 환경에서 더 높은 성능 향상을 확인 할 수 있었다.

향후에는 하이퍼레저 패브릭의 체인 코드 배치 처리 뿐만 아니라 트랜잭션이 검증 된 후 Ordering 을 위해 통신하는 구간 등의 하이퍼레저 패브릭의 데이터 셋 배치 처리를 연구할 예정이다. 네트워크 사용이 이뤄지는 구간을 배치 처리하여 네트워크 비용을 낮추게 되면 하이퍼레저 패브릭의 Throughput 과 Send Rate 의 개선이 이루어져 전체적인 성능개선이 이루어질 것으로 기대된다. 또한 본 논문에서 진행했던 체인 코드 메세지 배치 처리와 함께 사용한다면 독립적인 성능 향상으로 더 큰 폭의 개선이 이뤄질 것으로 기대된다.

ACKNOWLEDGMENT

이 성과는 2019 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2019R1G1A1100305).

참 고 문 헌

[1] Hyperledger Fabric. https://www.hyperledger.org/projects/fabric.

[2] Ethereum. https://ethereum.org.

[3] Bitcoin. https://bitcoin.org/en/.

[4] Chaincode. http://hyperledger-fabric.readthedocs.io/en/release- 1.1/ chaincode4noah.html. [Online; accessed 1-May-2018].

[5] Cloud Native Computing Foundation, “gRPC: A high performance, open-source universal RPC framework,” 2018. [Online]. Available: https://grpc.io/

[6] Docker. https://www.docker.com

[7] Hyperledger Caliper. https://www.hyperledger.org/projects/caliper.

[8] Christian Gorenflo, Stephen Lee, Lukasz Golab, Srinivasan Keshav“FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second”, 2019, arXiv.org

[9] Haris Javaid, Chengchen Hu, Gordon Brebner, “Optimizing Validation Phase of Hyperledger Fabric”, 2019, IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems(MASCOTS)

GAN 을 활용한 비트코인 불법거래 과표본 추출 기법

한정수°, 이채현, 고경찬, 우종수, 홍원기
포항공과대학교,정보통신대학원

{saw1515, chlee0211, kkc90, woojs, jwkhong}@postech.ac.kr

Bitcoin Illegal Transaction Oversampling Technique with GAN

Jungsu Han°, Chaehyeon Lee, Kyungchan Ko, Jongsu Woo, James Won-Ki Hong
Department of Computer Science and Engineering, POSTECH
Graduate School of Information Technology, POSTECH

요 약

비트코인 거래는 익명성이 보장되어 범죄와 관련된 불법거래 사례가 급증하고있다. 이를 해결하기 위한 연구들이 활발히 진행되고 있지만 불법거래를 수집하는 것과 그것을 정상거래와 구별하는 것이 어렵기 때문에 불법거래 표본의 갯수가 정상 거래 갯수보다 부족한 문제가 있다. 이와 같이 클래스 불균형 데이터셋을 다루는 방법으로 과표본추출(SMOTE) 모델을 사용하는 경우가 대다수이지만 해당 모델의 경우 데이터의 다양성을 충분히 표현하지 못할 가능성이 존재한다. 본 논문에서는 적대적 신경망(GAN)을 이용하여 비트코인 불법거래 탐지에 사용되는 데이터의 클래스 불균형 문제를 해결하는 방법을 제안한다. 그리고 생성된 데이터와 실제 데이터의 유사도를 검증하기 위해 XGBoost classification 을 활용하여 모델 성능을 정량적으로 평가하였다.

Ⅰ. 서 론

암호화폐란 암호화된 공개키를 이용하여 블록체인 네트워크상에서 거래가 가능하며, 해시 함수를 통해 쉽게 소유권을 증명할 수 있는 디지털 자산이다. 암호화폐의 핵심 기술인 블록체인은 분산장부(Distributed ledger)를 이용하여 개인간의 모든 거래 정보와 계좌 정보를 공개하지만 각각의 정보에 대한 신상은 제공하지 않음으로써 거래의 익명성을 보장한다[1]. 특히, 블록체인 네트워크에서 개인을 식별하는 것은 계좌의 주소만을 이용해서 가능한데 이 같은 계좌 주소를 개인이 여러 개를 생성하거나 소유 할 수 있다는 특징 때문에 익명성을 보장받게 된다. 하지만 이 같은 블록체인 거래의 익명성을 이용하여, 악의적인 목적을 가진 네트워크 참여자가 불법적인 거래를 일으킬 수 있는 문제가 있다. Sean[2]의 조사에 따르면 가장 대표적인 암호화폐인 비트코인의 경우 미・유럽을 중심으로 무기, 약물, 돈세탁 등을 목적으로 매년 76 억 달러에 해당하는 거래가 일어나고 있다고 분석하였다. 그렇기 때문에 블록체인이 차세대 금융시스템으로 발전하기 위해서는 불법・사기 거래들에 대한 탐지와 분류가 필수적이다.

분류 문제(classification problem)란 기존의 데이터 셋의 패턴을 분석하여 새로운 입력 데이터에 대한 클래스를 예측하는 문제이다. 일반적으로 기계학습을 통해 분류 모델을 개발하는데, 이때 학습할 데이터셋은 클래스의 갯수에 맞게 균등한 것이 이상적이지만 실제로 수집되는 대부분의 데이터의 경우 클래스가 불균형한 형태이다. 비트코인 불법 거래 분류문제에 사용되는 비트코인 거래 데이터의 경우에도 불법 거래(illegal transaction)의 주소를 확보하는 것이 어렵고 한정적이기 때문에 불법거래로 라벨링 된 거래보다 그렇지 않은 것이 훨씬 많이 존재하는 불균형 데이터셋이다. 대부분의 경우 불균형 데이터셋(imbalanced dataset)으로 인해 모델의 성능이 저하되는 것을 해결하기 위해 SMOTE(Synthetic Minority Over-sampling TEchnique)와 같은 리샘플링(resampling) 방법들을 사용하게 된다[3]. 그렇지만 SMOTE 의 경우 다른 클래스의 데이터와의 거리를 고려하지 않아 생성된 데이터의 클래스 중첩(overlapping of class)이 일어날 수 있는 문제가 있고 고차원 데이터(high dimensional data)에는 적합하지 않다. 본 논문에서는 이 문제를 해결하기 위해 적대적 생성 신경망(GAN, Generative Adversarial Network)을 이용하여 비트코인 불법거래 탐지에 사용되는 불균형 데이터 문제를 해결할 방안을 제시한다.

Ⅱ. 배경지식

1. Oversampling

Haixiang [4]은 대표적인 과표본추출 기법으로 Random Oversampling(ROS), SMOTE, ADASYN 가 사용된다고 설명하였다. ROS 는 소수 클래스 데이터 샘플을 무작위로 복제하여 생성하는 방식이고 SMOTE 는 데이터 샘플 간의 K-nearest neighbor 를 선택하여 그 점들을 이어 새로운 데이터 샘플을 만들게 된다. ADASYN 은 SMOTE 에서 발전된 기법으로 SMOTE 의 방식으로 만들어진 데이터의 왜곡된 분포를 막기 밀도분포(density distribution)을 이용하여 더 현실성 있는 표본을 생성하는 방식이다.

2. GAN and CGAN

GAN(generative adversarial network)은 기존의 데이터를 바탕으로 실제 데이터 셋에 없는 새로운

데이터를 생성하는 모델로 일반적으로 생성자(generator)와 구분 자(discriminator) 두 심층 네트워크로 이루어져있다[5, 6]. 생성자는 생성된 데이터에 랜덤 노이즈를 주어 생성된 데이터가 실제 데이터 분포와 유사한 형태를 만드는 것이 목적이고 구분자는 생성자가 만들어낸 데이터와 실제 데이터를 구별하는 것이 목적인 모델이다. 이 두 모델은 경쟁적(adversarial)으로 학습을 진행하여 더 현실적인 데이터를 생성하게 된다.

CGAN(conditional-GAN)은 GAN 의 확장된 모델로 data space 에 additional space 를 추가한 형태이다. 예를 들면 MNIST 데이터셋에서 특정 숫자의 값을 CGAN 으로 생성하고 싶으면 그 숫자의 label 을 additional space 로 추가하여 모델을 학습하면 된다.

3. WGAN and WCGAN

기존의 GAN 에서는 데이터 분포 간의 거리를 재는 방식으로 KL divergence 나 JS divergence 를 사용하였는데 이 두 방식은 생성 모델이 데이터의 특정 값(예를 들면 MNIST 의 label 이 1 인 이미지)에만 생성하는 mode collapse 현상이 발생하는 문제가 있었다. 이를 해결하기 위해 Arjovsky [7]은 실제 데이터의 확률분포를 알기 위해 Earth Mover(EM) distance 를 사용하였다. 해당 논문에서는 EM distance 값을 최소화하기 위해 Wasserstein-GAN(WGAN)을 제안하여 WGAN 이 생성자와 구분자의 균형에 예민한 GAN 이 학습 중에 발생할 수 있는 문제를 해결할 수 있음을 보였다.

Ⅲ. 실험

비트코인의 트랜잭션 데이터를 수집하기 위해 자체 비트코인 풀노드(full node)를 구축하였다. 그리고 불법 트랜잭션을 수집하기 위해 WalletExplorer.com나 해외 포럼과 같은 곳에서 불법 거래가 이루어진 트랜잭션의 hash나 지갑 주소를 조사하고 그에 해당하는 트랜잭션을 풀노드에서 수집하여 레이블링 하였다. 이 때 일반 거래소에서 발생한 트랜잭션을 0으로 레이블링 하였으며 silkroad(비트코인 불법거래 사이트)에서 발생한 트랜잭션을 1로 레이블링 하였다. 또한 수집한 트랜잭션의 특징(feature)을 알기 위해 트랜잭션의 비트코인 전송량, 전송 횟수, lifetime, fee, sibling 갯수 등을 함께 수집하였다. 수집한 특징에 대하여 주성분 분석(PCA)를 적용하여 15개의 주성분 벡터와 class 레이블 총 16개의 특징으로 데이터를 전처리 하였다. 그림 1은 전처리 한 주성분 벡터 중 데이터 분류에서 특징 중요도(feature importance)가 높은 상위 5가지를 시각화한 그래프이다.

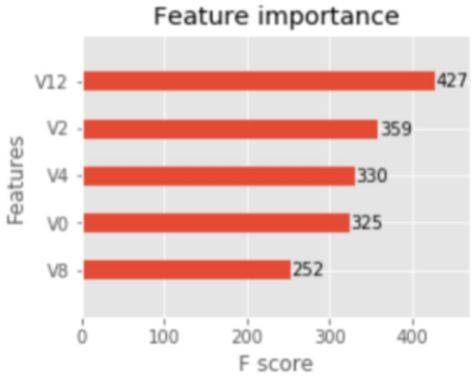


그림 1. Feature importance 상위 주성분 벡터

어떤 특징이 데이터 분류(classification)에 큰 영향을 주는지를 확인하기 위해 XGBoost algorithm[8]을 사용하였으며 분류에 영향력이 큰 상위 두 특징(PCA로 전처리된 V2, V12)을 이용해 GAN, WGAN 등으로 oversampling 데이터를 분석하였다. 또한 class 레이블을 추가한 CGAN, WCGAN을 구현하여 동일한 방식으로 분석하였다.

GAN 구조에 따른 생성 데이터는 그림 2와 같이 시각화 할 수 있다. 그림에서 초기 학습단계에서는 실제 데이터의 형태와 유사한 형태로 데이터를 생성하지만 학습 단계가 1000 step이 넘기 시작하자 일부 데이터 분포로 수렴하기 시작하였다. 이는 학습이 특정(non-optimal)한 샘플 분포에 수렴하는 mode collapse 때문으로 보인다. CGAN의 경우 class 값에 따라 각각 특정 분포로 수렴하게 된다. 데이터 간의 분포를 EM distance를 이용하여 분석하는 WGAN, WCGAN의 경우 GAN과 달리 class 정보 유무에 상관없이 모두 mode collapse를 보이지 않는 것으로 확인되었다.

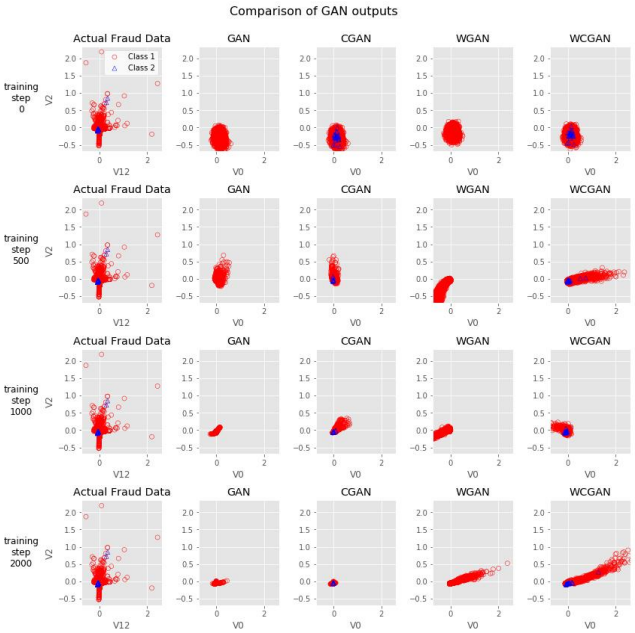


그림 1. GAN 종류에 따른 생성 데이터 시각화

GAN 모델 중 성능이 가장 좋게 나온 WCGAN과 SMOTE 방식으로 oversampling 된 각각의 데이터

셋을 이용하여 XGBoost classifier을 이용한 불법 트랜잭션 분류 결과를 정량적으로 비교하였다. 이때 학습되지 않은 WGAN 모델로 생성된 데이터와 실제 불법거래 데이터를 추가 하였을 때의 XGBoost classifier의 성능을 함께 표 1과 같이 정량적으로 분석하였다.

	auc	precision	recall	roc_auc
Untrained	0.9282	1.0	0.2826	0.9833
WCGAN	0.9280	0.9978	0.2808	0.9843
SMOTE	0.9291	0.9989	0.2919	0.9848
Real	0.9946	0.9693	0.9774	0.9993

표 1. XGBoost Classification 성능

WCGAN, Untrained WCGAN, SMOTE 방식으로 생성된 가짜 데이터를 활용하여 XGBoost classifier의 성능 변화를 테스트하였다. 만약 classifier의 성능에 변화가 있다면 생성된 데이터가 실제 데이터와 유사하다는 것을 의미한다. 그림 3을 보면 생성된 데이터가 추가됨에 따른 재현율(recall) 값의 변화 확인할 수 있다. Untrained WGAN의 경우 생성된 데이터가 추가됨에 따라 재현율의 변화가 크게 없는 것으로 확인되었다. 이는 생성된 데이터가 실제데이터와 유사하지 않다는 것을 의미한다. 하지만 WCGAN과 SMOTE 모델의 경우에도 생성된 데이터를 통한 클래스 분류의 재현율이 큰 차이가 나지 않는 것을 확인하였다. 생성된 데이터가 아닌 실제 데이터만을 사용하였을 경우 데이터 수에 따라 재현율이 0.9774까지 증가하는 것과 비교하였을 때 데이터 생성 모델(WCGAN, SMOTE)의 재현율 기준선(baseline)이 0.3 내외로 크게 차이 나는 것을 확인하였다

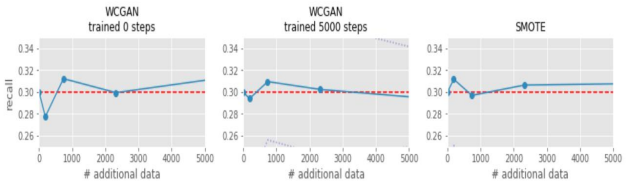


그림 3. WGAN vs SMOTE 비교

Ⅳ. 결론 및 향후 연구

본 연구는 비트코인 네트워크에서 생성되는 트랜잭션 데이터를 GAN 을 이용하여 과표본추출하는 방법을 제안하였다. 이를 통해 비대칭 데이터의 불균형 문제를 해결하여 기존의 과표본추출방식 보다 분류 모델의 성능향상이 있을 것으로 기대하였다. 그렇지만 실험결과 유의미한 성능 차이를 확인하지 못하였는데 이는 GAN 모델의 근본적인 문제와 트랜잭션 데이터 자체의 문제로 예상된다. GAN 모델은 구분자와 생성자 두 네트워크를 사용하게 되는데 이때 특정 네트워크의 성능이 지나치게 우수하면 다른 모델의 성능이 오르지 않는 현상이 일어날 수 있다. 또한 구분자가 실제 데이터 분포를 완벽하게 학습하면 GAN 의 기본 원리인 Minmax Game[8]에 모순이 생겨 학습 도중에 특정 데이터 분포에 진동(oscillation)하며 학습이 반복되는 문제가 있다. 그렇기 때문에 다른 기계학습 모델에 비해 학습이 최적의 값에 수렴하는 것이 상대적으로 어렵다. 그리고 수집한 트랜잭션 데이터의 특징 분포를 확인한 결과 불법거래와 정상 거래가 중첩되는 부분이 상당한 것으로 확인되어 클래스 분류 자체가 어렵다는 문제가 있었다. 향후 연구로는 최근 우수한 성능을 보여주고 있는 DCGAN 을 이용하여 모델의 성능을 높이고 데이터의 클래스를 분류에 영향을 줄 수 있는 추가적인 특징들을 수집하여 데이터클래스 분류가 더욱 용이하도록 할 예정이다

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 임 (No.2018-0-00539)

참 고 문 헌

[1] Nakamoto, S. Bitcoin: A peer-to-peer Electronic Cash System (2008)

[2] Foly, S. Karlsen, J. R., & Putniņ ,T. J. Sex, Drugs, and Bitcoin: How Much Illegal Activity Is Financed through Cryptocurrencies?. The Review of Financial Studies, pp. 1798- 1853. (2019)

[3] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. SMOTE: synthetic minority oversampling technique. Journal of artificial intelligence research. pp. 321-357. (2002)

[4] Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. Learning from class-imbalanced data: Review of methods and applications. Expert Systems with Applications. pp. 220-239. (2017)

[5] Mullick, S. S., Datta, S., & Das, S. Generative adversarial minority oversampling. In Proceedings of the IEEE International Conference on Computer Vision. pp. 1695-1704. (2019)

[6] Ba, H. Improving Detection of Credit Card Fraudulent Transactions using Generative Adversarial Networks. arXiv preprint arXiv:1907.03355. (2019).

[7] Arjovsky, M., Chintala, S., & Bottou, L. Wasserstein gan. arXiv preprint arXiv:1701.07875. (2017).

[8] Chen, T., & Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785-794. (2016)

[9] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S. & Bengio, Y. Generative adversarial nets. In Advances in neural information processing systems. pp. 2672-2680. (2014)

[10] Walletexplorer: smart bitcoin block explorer. Available at https://www.walletexplorer.com/.

[11] Bitcoin.org. Bitcoin core json apis. Available at https://bitcoin.org/en/developer-reference#bitcoin-core-apis.

Q-Learning 기반 VNF 자원 인식 Service Function Chaining

이도영, 유재형, 홍원기

포항공과대학교 컴퓨터공학과

{dylee90, jhyoo78, jwkhong}@postech.ac.kr

Q-Learning based VNF Resource-aware Service Function Chaining

Doyoung Lee, Jae-Hyoung Yoo, James Won-Ki Hong

Department of Computer Science and Engineering, POSTECH

요 약

오늘 날 5G 네트워크 시대가 도래 하면서, 급변하는 서비스 요구사항을 만족시키기 위해 유연하고 민첩하게 네트워크를 구축하고 관리하는 것이 요구되고 있다. 소프트웨어 정의 네트워킹 (SDN, Software-Defined Networking)과 네트워크 기능 가상화 (NFV, Network Function Virtualization)는 네트워크를 소프트웨어 기반으로 전환하여 유연한 네트워크 관리를 가능케 하는 핵심 기술들이다. 특히, NFV는 네트워크 기능들을 소프트웨어 형태로 가상화하여 상용 서버에서 운영하며, 동적으로 네트워크 기능을 관리할 수 있는 장점이 있다. 하지만, 한편으로는 물리 네트워크 자원 뿐 아니라 수 많은 가상 네트워크 및 자원들로 인해 네트워크 관리를 복잡하게 만드는 원인이 된다. 이를 위해 최근에는 인공지능 기술을 도입하여 복잡한 NFV 환경 및 기술들을 관리하는 연구가 주목받고 있다. 특히, 서비스 펑션 체이닝 (SFC, Service Function Chaining)은 필수 NFV 기술 중 하나로, 효율적인 SFC를 생성하는 것이 요구된다. 본 논문에서는 인공지능 기술 중 하나인 강화학습 (RL, Reinforcement Learning)을 활용하여 VNF 자원 사용량을 고려한 최적 SFC 경로를 찾는 방법을 제안한다.

Ⅰ. 서 론

네트워크 기능 가상화 (NFV, Network Function Virtualization)는 전용 하드웨어를 통해 제공되는 네트워크 기능을 소프트웨어 형태로 가상화하여 상용 서버에서 운영하는 기술이다. 5G 네트워크에서 NFV는 소프트웨어 정의 네트워킹 (SDN, Software-Defined Networking)과 함께 신속하고 유연하게 네트워크를 관리하여 서비스를 제공할 수 있는 핵심 기술로 활용되고 있다. NFV는 서비스 요구사항에 따라 동적으로 네트워크 기능을 VNF (Virtual Network Function) 형태로 배치할 수 있지만, 한편으로는 수 많은 가상 네트워크와 자원들을 관리하게 만들어, 네트워크 관리를 복잡하게 만드는 원인이 된다. 또한, 서비스 요구사항이 급변하고 다양한 기기들이 네트워크에 연결되는 5G 네트워크 시대에, 네트워크는 더욱 복잡해지고 있으며 이는 관리자에 의한 네트워크 운영 및 관리를 어렵게 만들고 있다.

복잡해지는 네트워크 관리 문제를 해결하기 위해 최근에는 인공지능 기술을 네트워크 관리에 접목하려는 연구가 주목받고 있다. 기존에는 주로 기계학습 알고리즘들을 침입 탐지, 비정상 트래픽 분류 등을 위해 활용하였지만, 오늘 날에는 다양한 인공지능 기술들을 VNF 배치 (VNF Deployment), 서비스 펑션 체이닝 (SFC, Service Function Chaining), 오토 스케일링 (Auto-Scaling), 마이그레이션 (Migration) 등 전반적인 NFV 관리 기술에 적용하기 위한 연구가 진행되고 있다. 그 중에서도 SFC는 네트워크 트래픽에 일련의 네트워크 기능을 적용할 수 있도록 트래픽 경로를 제어해 VNF을 경유시키는 기술로써, QoS를 보장하는 최적의 SFC 경로를 선택하는 것이 요구된다.

SDN/NFV를 활용해 최적의 SFC 경로를 찾는 방법에 대한 많은 연구가 진행되어 왔지만, 동적으로 변화하는 네트워크 환경에 대응하는 SFC 생성 방법에 대해서는 아직 많은 연구 이슈가 존재한다. 이를 위해 본 논문에서는 인공지능 기술 중 하나인 강화학습 (RL, Reinforcement Learning)을 활용하여 SFC를 구성하는 VNF들의 현재 자원 사용률과 VNF 설치 위치를 고려한 SFC 경로 선택 방법을 제안한다. 제안하는 방법은 OpenStack 기반 테스트베드에서 실제 VNF들을 설치한 후, 임의로 선택한 SFC 경로보다 강화학습 기법 중 하나인 Q-Leraning을 사용해서 SFC 경로를 선택했을 때, 패킷 응답 시간 측면에서 더 효과적인 경로인 것을 검증하였다.

Ⅱ. 관련 연구

최근까지 인공지능 기술을 SDN/NFV 기반 네트워크 환경의 효율적인 관리를 위해 활용하는 연구가 활발하게 진행되고 있다. 그 중, [1]에서는 대용량 데이터 전송 트래픽이 흐르는 SDN 환경에서 네트워크 혼잡 (Congestion)을 피하기 위해 Flow 경로 설정에 강화학습 Q-Learning 알고리즘을 활용하였다. 또한, [2]에서는 강화학습 중 Temporal Difference (TD) 알고리즘을 Q-value에 적용해 최적의 SFC 경로를 선택하는 방법을 제안하였다. 이 때, SFC를 구성하는 SF (Service Function)의 현재 CPU 사용량과 대역폭 (Bandwidth)을 고려해서 보상 (Reward) 값을 책정하였으며, 보상이 가장 높은 SF들을 선택하여 SFC를 구성하였다. 그 외 [3]에서는 Q-Learning을 사용하여, SFC를 구성하는 SF를 물리 노드 (Physical Node)에 분산 배치시키는 Load-balancing 방법을 제안하였다. 마지막으로 [4]에서는 지도학습 (Supervised Learning) 알고리즘인 RFR (Random Forest Regression)과 FNN (Feed-forward Neural Network)으로 SFC를 구성하는 링크 상태 (Link State)를 학습하여 SFC를 통한 Throughput을 예측하는 방법을 제안하였다.

위 연구들은 인공지능 기술을 SFC 생성에 활용할 수 있는 방향을 제시하였지만, 검증을 위한 실험 환경은 Mininet 또는 자체적으로 구현한 시뮬레이션을 통해 구축되었다는 한계가 존재한다. 한편, ETSI에서는 NFV 구조와 SFC 기능 표준화도 지속적으로 추진하고 있는데, [5]에서는 SFC 기능을 제공하는 ETSI 표준에 부합하는 프레임워크가 없음을 지적하고 있다. 따라서 강화학습을 활용한 향후 SFC 연구들은 최적의 SFC를 생성하는 문제 뿐 아니라 ETSI 표준 프레임워크에서 동작할 수 있는 형태로 구현하는 것도 고려할 필요가 있다.

Ⅲ. Q-Learning 기반 SFC 경로 선택

강화학습은 인공지능 기술 중 하나로, 최대의 누적 보상을 주는 행동(Action)들을 수행할 수 있도록 시행착오를 거쳐 최적의 정책 (Policy)을 찾는 학습 방법이다. 일반적으로 강화학습은 에이전트 (Agent)와 환경 (Environment)으로 구성되며, 에이전트는 정책에 따라 현재 상태 (State)에서 특정 행동을 수행하게 된다. 이를 통해 다음 상태로 이동하게 되며, 동시에 보상을 얻게 된다. 강화학습의 목적은 각 상태에서 다양한 행동을 수행하며 보상을 최대화하는 정책을 찾는 것이기 때문에 상태와 행동, 그에 대한 보상의 정의가 필요하다.

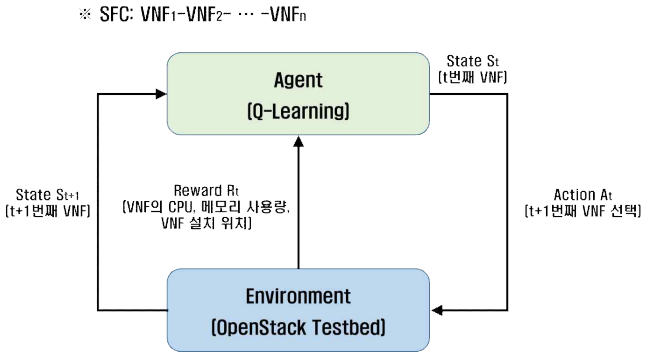


그림 1 강화학습 기반 SFC 경로 선택 모델

본 논문에서 제안하는 강화학습 기반 SFC 경로 선택 방법은 [그림 1]처럼 미로 찾기와 유사한 방법으로 단순화하여 문제를 정의한다. 미로 찾기 문제에서는 현재 미로 내 위치를 상태로 정의하며, 어느 방향으로 이동할 것인지 행동으로 정의한다. 그리고 시행착오를 거쳐 목적지에 도달하게 되면 미로를 탈출했다는 보상을 준다. 제안하는 강화학습 기반 SFC 경로 선택 문제도 이와 유사하게 SFC를 구성하는 각 VNF를 패킷이 통과해야 하는 상태로 정의하고, 다수의 VNF 중 어떤 것을 다음 VNF로 선택할지를 행동으로 정의한다. 그리고 미로에서 탈출구에 도달하는 것처럼 SFC를 구성하는 VNF들을 순서대로 올바르게 선택했을 때 보상을 준다. 다만, SFC 경로 선택에서는 단순히 VNF들을 순서대로 선택한 것만으로 보상을 주는 것이 아니라, 최적의 성능을 제공하는 VNF를 선택해 SFC 경로를 구성 했는지 여부도 가중되어 보상으로 고려되어야 한다. 본 논문에서는 SFC를 구성할 때, 각 VNF들의 현재 CPU 사용량 및 메모리 사용량, 그리고 각 VNF가 설치 된 물리 노드의 위치를 고려하여 보상으로 책정한다. CPU와 메모리는 패킷 처리 성능에 영향을 미치는 자원들로, 각 자원들이 충분치 않으면 패킷 처리가 지연 (Delay)되거나 이로 인해 패킷 손실(Packet loss)이 발생할 수 있다 [6, 7]. 또한, VNF가 설치 된 물리 노드 위치는 SFC 경로를 지나는 패킷 전달 시간과 간접적으로 관련이 있는 요소이다. SFC를 구성하는 VNF들이 같은 물리 노드에 위치해 있을 경우, 패킷들은 다른 물리 노드로 전달 될 필요 없이 동일한 물리 노드 내에서만 이동하기 때문에 VNF 간 패킷 전달 시간이 감소한다.

오늘 날 존재하는 여러 가지 강화학습 알고리즘 중, 본 논문에서는 Q-Learning을 최적의 SFC 경로를 선택하는 데 활용한다. Q-Learning은 마르코프 결정 과정 (MDP, Markov Decision Process)의 최적 정책을 찾기 위해 활용할 수 있는 강화학습 방법이다. 이 때, 현재 상태에서 특정 행동을 수행할 때 얻을 수 있는 보상을 예측하는 지표인 Q값을 반복적으로 학습하여 특정 상태에서 어떤 행동을 수행하는 것이 좋을지 나타내는 정책으로 사용한다. 제안하는 방법에서 Q-Learning의 Q값 갱신 과정은 수식 (1)과 같이 표현할 수 있다. SFC 경로 선택 문제에서 S는 패킷이 지나가는 SFC의 VNF를 의미하며, A는 다음 VNF로 이동하는 행동을 의미한다. R은 특정 상태에서 선택한 행동으로 인해 받을 수 있는 보상, 즉 선택된 다음 VNF의 자원 사용량과 물리 노드 위치에 따라 책정 된 보상 값을 의미한다. 이 외에 η은 학습률 (Learning rate), γ은 할인율 (Discount factor)을 의미한다. 결과적으로, 제안하는 방법은 수식 (1)에 따라서 자원 사용량 및 물리 노드 위치를 고려해 최적의 SFC를 구성할 수 있는 VNF들을 선택해 경로를 생성하는 것이다.

$$Q(S_t, A_t) = \eta(R + \gamma \max(Q(S_{t+1}, A)) \quad (1)$$

또한, Q-Learning 수행 중, 특정 상황에서 행동을 선택할 때, 어떤 비율로 탐사 (Exploitation)와 탐험 (Exploration)을 결정할지 고려하는 것이 필요하다. 일반적으로 탐사는 특정 경우를 우선시하여 정책을 찾는 방식이고, 탐험은 모든 경우를 고려하여 최적의 정책을 찾기 위한 방식이다. 즉, Q-Learning에서 Q값이 가장 높은 행동을 선택하는 것이 탐사이며, 임의로 다른 행동을 선택하는 게 탐험이다. 본 논문에서는 학습 초기에 탐험 비중을 더 중시해 다양한 VNF를 선택하여 최적 SFC 경로를 찾는데, 이를 위해 ε-greedy 알고리즘을 활용한다. ε-greedy 알고리즘은 ε값을 기준으로 탐험과 탐사 중 어떤 것을 수행할지 결정하는 것이며, ε값이 클수록 탐험을 선택할 확률이 커진다. 제안하는 방법에서는 ε값을 매 학습 때마다 일정 비율을 감소시켜서, 탐험을 할 확률을 점차 낮춘다.

알고리즘 1. Q-Learning 기반 SFC 경로 선택

초기화 (Initialization) 과정:
- Q-value ← 모든 Q-value 0으로 초기화
- R-value ← 각 State를 선택할 때 받는 보상
  - ※ CPU 및 Memory utilization, VNF 설치 Node 위치로 보상 책정
- (η, γ, ϵ) ← 변수 초기화 ※ (학습률, 할인율, ε-greedy 값)

Q-Learning 과정:
- iteration_num ← n
- Episode ← 1
- **while** (Episode ≤ iteration_num) ※ n번 학습 반복
  - ε ← ε-greedy 값 일정 비율 감소
  - **while** (SFC path is not decided) ※ SFC 경로 선택
    - A_t ← a ※ 현재 상태 S_t에서 수행할 행동 결정
    - S_{t+1} ← (S_t, A_t) ※ S_{t+1}로 이동하는 다음 VNF 선택
    - Q-value ← 새로운 Q-value 갱신 ※ 수식 (1)
  - Episode ← Episode + 1

SFC 생성 과정:
- SFC_path ← S_0부터 Q-Value가 큰 행동을 선택, VNF 결정

본 논문에서 제안하는 Q-Learning 기반 SFC 경로 선택 알고리즘의 전체적인 수행과정은 알고리즘 1과 같다. 학습에 앞서, 특정 상황에서 어떤 행동을 선택할지 결정하는 정책인 Q값은 0으로 초기화하며, 현재 실행 중인 각 VNF들의 자원 정보 (CPU 및 메모리 사용량)와 배치된 물리 노드 위치를 환경 (Environment)로부터 받아온 후 보상 값으로 책정하여 저장한다. 또한, 학습 과정에서 사용되는 각 변수의 초기 값도 설정한다. 다음 순서로는 iteration_num로 정의 된 횟수만큼 SFC 경로를 찾아가는 과정에서 Q-Learning을 반복하며 Q값을 갱신하는 것이다. 학습 과정에서 초기 ε-greedy 값은 큰 값을 할당하여 학습이 반복될 때마다 일정 비율로 감소시킨다. 이는 학습 초기에는 Q값에 근거한 행동 선택보다 임의의 행동을 선택하는 탐험 위주로 다양한 VNF를 선택하며 이에 대한 보상을 얻기 위해서다. 이 후, 학습이 일정 수준 반복되었을 때에는 ε값이 초기 값보다 작아졌기 때문에 탐험보다는 Q값에 근거하여 최적의 행동을 선택하게 된다. 모든 학습을 마친 후에는 계산된 Q값들을 활용하여, 초기 상태 (S_0), 즉, SFC를 경유해야 하는 패킷이 트래픽 분류기 (Traffic Classifier)에 도착했을 때부터 어떤 VNF들로 흘러야 하는지 SFC 경로를 결정한다. 최종 결정 된 SFC 경로는 실제 네트워크 환경에 SFC로 구성하여 Q-Leaning 기반 SFC 경로 선택 및 생성까지 마무리된다.

Ⅳ. 성능 평가

Q-Learning 기반 SFC 경로 선택 방법의 성능 검증을 위해 [그림 2]와 같이 OpenStack 기반으로 테스트베드를 구축하였다. 테스트베드는 OpenStack 환경을 관리하는 컨트롤러 노드 (Controller Node)와 VNF이 설치 될 수 있는 4개의 컴퓨트 노드 (Compute Node)로 구성된다. 테스트베드 내에 VNF를 설치 할 때에는 임의의 컴퓨터 노드를 선택하여 VNF가 설치 된 VM 인스턴스를 생성하며, 각 VM 인스턴스에는 하나의 VNF만 동작한다. 본 성능 평가에서는 모든 VM 인스턴스들을 미리 테스트베드에 생성해 놓은 상태에서 SFC를 구성하는 VM 인스턴스들을 선택해 SFC를 구성한다. 또한, 테스트베드 내 위치한 모니터링 노드 (Monitoring Node)는 실시간으로 테스트베드 내 생성된 VNF의 자원 사용량 정보를 모니터링하여 시계열 데이터베이스에 저장한다.

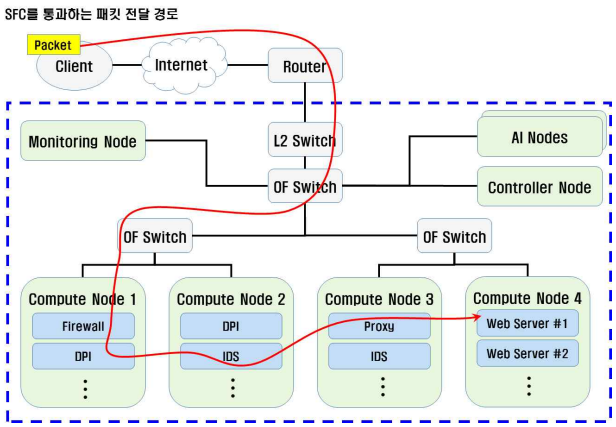


그림 2 강화학습 기반 SFC 성능 평가 테스트베드

성능 측정 시나리오는 [그림 2]의 클라이언트 (Client)에서 다수의 HTTP 요청 메시지 (HTTP Request Message)를 발생시켜 테스트베드 내 SFC 경로를 경유해 웹 서버로 전달 한 후, 요청 메시지에 대한 모든 응답 메시지 (Response Message)를 받는 데 소요되는 시간을 측정하였다. 이 때, HTTP 요청 메시지 생성은 Apache 웹 서버 상태를 측정하는 성능 측정 도구인 AB (Apache HTTP Server Benchmarking tool)을 사용하였다. 웹 서버로 HTTP 요청 메시지가 전달되기 전에 경유하는 SFC는 4개의 VNF (Firewall, DPI, IDS, Proxy)로 구성된다. 방화벽 기능을 하는 Firewall VNF는 Iptables를 사용하였으며, DPI는 nDPI [8], IDS는 Suricata [9], 그리고 Proxy 기능을 제공하는 VNF는 HAProxy [10]를 설치하였다. 마지막으로, HTTP 요청 메시지에 대한 응답을 위해 두 개의 Apache 웹서버를 설치하고, SFC의 마지막 VNF인 HAProxy가 Round-robin으로 각 웹 서버로 패킷을 전달하도록 설정하였다. 또한, 선행연구 [4]와 동일하게 각 VNF 별로 3개의 VM 인스턴스를 생성하여 총 12개의 VM 인스턴스가 테스트베드 내에 존재한다. 이 때, VNF 자원 사용량에 의해 패킷 처리 성능이 쉽게 영향 받을 수 있도록 VNF가 동작하는 각 VM 인스턴스에 작은 양의 자원 (vCPU 1개, 메모리 512MB, 디스크 20GB)을 동일하게 할당하였다. 모든 VM 인스턴스 생성을 마친 후에 VNF들을 선택해 생성할 수 있는 SFC 경로 개수는 [그림 3]과 같이 81 (3^4)개이다.

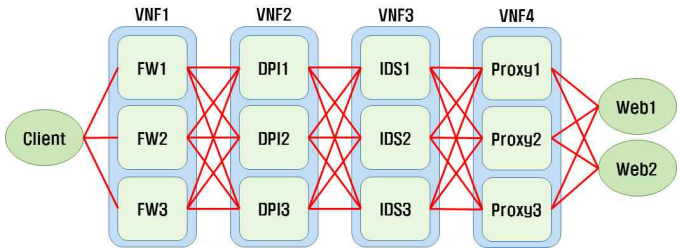


그림 3 테스트베드 내 선택 가능한 SFC 경로

VNF 자원 상태를 고려한 Q-Learning 기반의 SFC 경로 선택의 성능 평가를 위해, 동일한 성능 측정 시나리오에서 임의로 SFC를 구성한 경우의 성능과 비교하였다. 이 때, 성능 측정 과정에서 Stress-ng [11] 도구를 사용하여 임의의 VNF에서 CPU 및 메모리를 사용하도록 설정하였다. 임의로 SFC 경로를 선택하는 경우에는 각 VNF에서 소모하고 있는 자원을 고려하지 않고, 단순히 SFC를 구성하는 VNF들의 순서만을 고려하여 경로를 선택한 후 SFC를 생성한다. 반면, Q-Learning 기반의 SFC 생성은 현재 측정된 VNF의 자원 사용량 (CPU, 메모리), 그리고 다음 순서의 VNF와 동일한 물리 노드에 배치되어 있는 지 여부를 보상으로 고려하여 학습을 수행한다. 학습은 테스트베드 내 설치된 AI 노드에서 수행되며, 보상을 책정하기 위해 필요한 VNF 자원 사용량 및 물리 노드 위치 정보는 모니터링 노드로부터 받아온다. 마지막으로, Q-Learning 학습에 사용한 초기 변수 값은 [표 1]과 같다.

표 1. Q-Learning 초기 변수 정보

변수 종류	값
η (학습률)	0.1
γ (할인율)	0.6 ~ 0.9
ϵ (탐험 확률)	0.99
iteration_num (반복 횟수)	1000 ~ 3000

학습률 η의 초기 값은 0.1로 고정하여 사용하였으며, 미래의 보상을 얼마나 가중치를 두고 환산할지 결정하는 시간할인율 γ 값은 0.6에서 0.9 사이의 값으로 변경하면서 성능 평가를 진행하였다. 또한, ε-greedy 알고리즘의 초기 ϵ 값은 0.99로 설정하여, 초반 학습 과정에서는 대부분의 경우 Q값에 따른 행동 선택보다는 임의의 행동을 선택하도록 유도하였다. ϵ 값은 학습을 반복할수록 일정 비율로 줄어들게 된다. 마지막으로, 학습 반복

횟수는 1000회부터 3000회까지 임의로 수행하였다. 초기 ϵ 값을 높게 설정하여, 학습 반복 횟수가 작을 경우에는 탐험만으로 Q값을 갱신하게 되기 때문에 학습 횟수는 적절히 탐험과 탐사를 수행하며 Q값을 학습할 수 있도록 정하는 것이 필요하다. 본 성능 평가에서 1000회 이상으로 학습을 진행해본 결과, 4개의 VNF로 구성된 SFC 경로를 찾는 데에는 1000회 이상 학습으로 최적의 SFC 경로를 선택할 수 있음을 확인하였다.

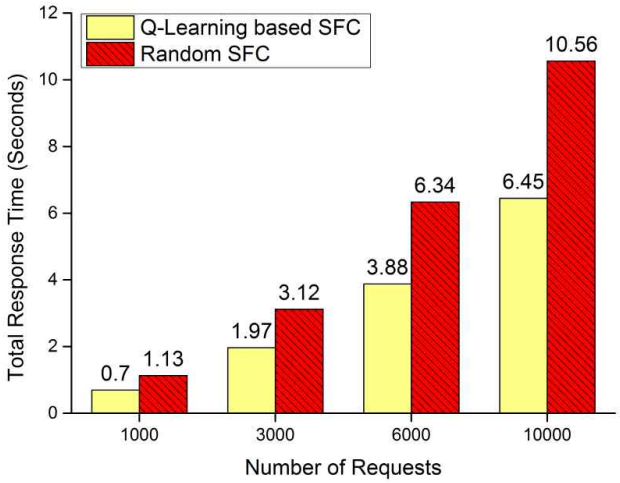


그림 4 SFC를 통한 HTTP 요청 메시지의 응답 시간

모든 설정을 마치고 나서, 서로 다른 방법으로 생성된 두 개의 SFC로 HTTP 요청 메시지를 1,000개, 3,000개, 6,000개, 그리고 10,000개를 발생시킨 후, 모든 응답 메시지가 클라이언트로 도착하는 시간 (Total Response Time)을 측정한 결과는 [그림 4]와 같다. 전반적으로, 임의로 각 VNF를 선택해 SFC를 구성하는 경우에는 Q-Learning 기반으로 생성한 SFC에 비해 응답 메시지를 모두 받을 때까지 더 긴 시간이 필요하였다. 또한, HTTP 요청 메시지 개수가 증가할수록 두 SFC에서 모든 응답 메시지를 받는 데 필요한 시간도 증가하지만, 임의로 생성한 SFC의 경우에 증가 폭이 더 큰 것을 확인할 수 있었다. 결과적으로, Q-Learning으로 생성한 SFC는 임의로 생성한 SFC에 비해 같은 HTTP 요청 메시지 개수일 때, 응답을 받는 데 소요되는 시간 대비 61% 정도의 시간이 소요된다. 이를 통해, 제안하는 Q-Learning 기반 SFC 경로 선택 방법은 각 VNF들의 자원 사용량과 설치된 물리 노드 위치를 보상으로 활용하여 효과적으로 SFC 경로를 선택할 수 있음을 확인하였다.

V. 결론 및 향후 연구

본 논문에서는 강화학습 알고리즘 중 하나인 Q-Learning을 활용해 SFC 경로를 찾는 방법을 제안하였다. 제안하는 방법은 SFC를 구성하는 VNF의 현재 CPU 사용량과 메모리 사용량, 배치된 물리 노드 위치를 고려하여 보상을 책정하고, 이를 기반으로 짧은 HTTP 응답 시간을 제공하는 SFC 경로를 선택한다. 제안하는 Q-Learning 기반 SFC 경로 선택 방법은 임의로 SFC 경로를 선택하는 것보다는 응답 시간 측면에서 좋은 경로를 선택할 수 있도록 돕지만, 현재 자원 상태와 VNF 설치 위치 같은 제한된 정보만을 보상으로 고려한 경로 선택이기 때문에 향후 네트워크 혼잡 등 동적으로 네트워크 상태가 변할 때 대응하기 어렵다는 문제가 있다. 또한, 미로찾기와 유사하게 단순한 경로 선택 문제로 모델링하였기 때문에 SFC 성능에 영향을 미치는 다양한 경우를 상태로 정의하지 못했다는 한계가 존재한다. 따라서 향후 연구로는 단순한 Q-Learning이 아닌, 신경망을 도입한 DQN (Deep Q Network) 등의 기법을 활용하여 NFV 환경에서 고려할 수 있는 다양한 경우들을 상태로 정의한 후, SFC 생성 연구에 활용할 예정이다. 또한, 최적의 SFC 경로를 찾기 위해 요구되는 학습 시간, 다른 최적화 기법과의 비교 등 다양한 시나리오에서의 성능 검증을 진행할 계획이다.

ACKNOWLEDGMENT

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발)

참 고 문 헌

[1] Kim, Seonhyeok, Jaehyeok Son, Ashis Talukder, and Choong Seon Hong. "Congestion prevention mechanism based on Q-leaning for efficient routing in SDN." In 2016 International Conference on Information Networking (ICOIN), pp. 124-128. IEEE, 2016.

[2] Ku, Hye-Jin, J. H. Jung, and Gu-In Kwon. "A study on reinforcement learning based SFC path selection in SDN/NFV." International Journal of Applied Engineering Research 12, no. 12 (2017): 3439-3443.

[3] Kim, Sang Il, and Hwa Sung Kim. "A research on dynamic service function chaining based on reinforcement learning using resource usage." In 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 582-586. IEEE, 2017.

[4] Jeong, Seyeon, Heegon Kim, Jae-Hyoung Yoo, and James Won-Ki Hong. "Machine Learning based Link State Aware Service Function Chaining." In 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1-4. IEEE, 2019.

[5] Mechtri, Marouen, Chaima Ghribi, Oussama Soualah, and Djamal Zeghlache. "NFV orchestration framework addressing SFC challenges." IEEE Communications Magazine 55, no. 6 (2017): 16-23.

[6] Gallenmüller, S., Emmerich, P., Wohlfart, F., Raumer, D., & Carle, G. (2015, May). Comparison of frameworks for high-performance packet IO. In 2015 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS) (pp. 29-38). IEEE.

[7] Dobrescu, Mihai, Norbert Egi, Katerina Argyraki, Byung-Gon Chun, Kevin Fall, Gianluca Iannaccone, Allan Knies, Maziar Manesh, and Sylvia Ratnasamy. "RouteBricks: exploiting parallelism to scale software routers." In Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, pp. 15-28. 2009.

[8] Deri, Luca, Maurizio Martinelli, Tomasz Bujlow, and Alfredo Cardigliano. "ndpi: Open-source high-speed deep packet inspection." In 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 617-622. IEEE, 2014.

[9] Open Information Security Foundation, "Suricata: Open Source IDS/IPS/NSM engine", [Online]. Available: https://suricata-ids.org/

[10] HAProxy, "HAProxy: The Reliable, High Performance TCP/HTTP Load Balancer", [Online]. Available: http://www.haproxy.org/

[11] King, Colin Ian. "Stress-ng." URL: http://kernel. ubuntu. com/git/cking/stressng. git/(visited on 28/03/2018) (2017)

컨테이너 기반의 클라우드 상에서 ScienceDMZ 기술 적용을 통한 데이터 전송 성능 향상

이상권, 석우진*, 문정훈*, 김기현*, 장민석*, 김천용*,
과학기술연합대학원대학교, *한국과학기술정보연구원

sglee@kisti.re.kr, *wjseok@kisti.re.kr

Data Transfer Performance Improvement through Application of ScienceDMZ Technology in Container-Based Cloud

Lee Sang Gwon, Seok Woo Jin*, Moon Jeong Hoon*, Kim Ki Hyeon*, Jang Min Seok*,
Kim Cheon Yong*
UST., *KISTI.

요 약

최근 컴퓨팅 자원들을 효율적이고 유연하게 사용할 수 있는 가상화 기술이 뜨고 있다. 특히 가상화 기술중에서 컨테이너 기반의 가상화는 이전 하이퍼바이저를 사용하는 가상화에 비해 가볍고 빠르며 확장이 쉽다는 장점을 가진다. 현재 컨테이너 기반의 클라우드로 쿠버네티스(Kubernetes)가 많이 활용되고 있다. 쿠버네티스에서 데이터 전송 소프트웨어가 저장되어 있는 이미지로 파드를 생성해 파일을 전송하는데 활용할 수 있는데, 이 경우 쿠버네티스 클러스터의 노드가 일반적인 성능의 서버와 1Gbps 의 네트워크를 사용한다면 기본적인 설정만으로도 성능을 모두 사용할 수 있지만 노드로 사용되는 서버가 고성능의 하드웨어를 사용하고 10Gbps 이상의 네트워크를 사용한다면 제 성능을 내기 어렵다는 문제점이 있다. 이러한 이유로 본 논문에서는 과학 빅데이터 전송 공유망 구성에 사용되는 기술인 ScieceDMZ 를 적용하여 컨테이너 기반의 클라우드에서 데이터 전송 성능을 최대한 이끌어 낼 수 있도록 하고자 한다. ScienceDMZ 는 방화벽 우회, 망분리, 네트워크 튜닝 등의 기술을 사용하여 전송 성능을 높이는 기술이다. 기존 데이터 전송 성능과 ScienceDMZ 기술 적용 후 전송 성능을 비교하였고, 그 결과 전송 성능이 향상되었음을 확인하였다.

Ⅰ. 서 론

현재 클라우드 분야에서 컨테이너 기술이 등장하면서 뛰어난 편의성과 성능 덕분에 사용률이 증가되고 있는 추세이다. 이전 클라우드에서 많이 사용되었던 방식인 베어메탈에 비해 가상화 기술 중 하나인 컨테이너 방식은 기존 베어메탈 기반의 클라우드가 노드 단위로 사용자에게 할당해주는데 비해 CPU, 메모리, GPU 등의 자원을 사용자의 요구치 만큼 분할하여 할당해주는 것이 가능하다. 덕분에 컨테이너 기반의 클라우드에서는 하나의 노드로 다수의 사용자가 나누어 사용할 수 있어 낭비되는 자원이 적어 보다 효율적으로 동작한다.[1-2] 대표적인 컨테이너 기반 클라우드에는 쿠버네티스가 있다.[3] 쿠버네티스에서는 어플리케이션과 어플리케이션 실행을 위한 환경이 저장되어 있는 도커 이미지를 사용해 컨테이너를 생성해준다. 쿠버네티스는 다양한 어플리케이션을 종속성 문제나 복잡한 설치 과정 없이 실행할 수 있고 GPU 같은 고성능 자원을 필요로 하는 경우에도 쉽게 확보할 수 있다는 장점을 가지고 있다. 이러한 장점을 가진 쿠버네티스를 초고속 데이터 전송망 구축에도 활용하고자 한다.

쿠버네티스 클러스터에서 데이터 전송 소프트웨어의 이미지를 통해 쿠버네티스의 실행 단위인 파드를 생성하면 바로 파일을 주고받는게 가능한 환경이 만들어진다. 여기서 일반적으로 사용되는 1Gbps 의 네트워크를 사용해서 전송한다면 제 성능을 내는데 별 문제가 없지만 10Gbps 이상의 네트워크 대역폭을 사용하게 되면 기본적인 환경만으로는 성능을 최대로 끌어올리기가 힘들다

는 문제가 있다. 본 논문에서는 이러한 문제점을 해결하기 위해 여기에 현재 진행되고 있는 초고속 데이터 전송망 구축 프로젝트에서 사용되고 있는 방법들을 가져와 적용해 봄으로써 전송 속도의 변화를 비교 분석하였다. 그 결과 기존에 비해 전송 속도가 향상되었고 클라우드 상에서도 초고속 데이터 전송망 구축이 가능함을 확인하였다.

논문의 구성은 2 장에서는 컨테이너 기반 클라우드 시스템과 고대역폭 전송망 구현 기술인 ScienceDMZ 대해서 설명한다. 3 장에서는 클라우드 내에서 ScienceDMZ 의 주요 설정들을 조절하면서 적용 전과 적용 후를 비교하는 실험을 진행한다. 그리고 4 장에서 끝을 맺는다.

Ⅱ. 컨테이너 기반 클라우드와 ScienceDMZ 기술

이번 장에서는 노드가 분산되어 배치되어 있는 환경에서 구현된 컨테이너 기반 클라우드 시스템과 네트워크 구성에 대해 설명하고 초고속 전송망 구현 기술인 ScienceDMZ 에 대해 소개한다. 그리고 ScienceDMZ 기술을 앞에서 소개한 클라우드 시스템에 활용할 수 있는 방안에 대해 소개한다.

2.1 컨테이너 기반 클라우드 시스템

컨테이너 기반 클라우드 시스템 구축에는 도커 스웜, 메소스, 쿠버네티스가 주로 사용된다. 본 논문에서는 이 중 다양한 기능을 지원하고 있는 쿠버네티스를 사용해 클라우드를 구축하였다. 쿠버네티스는 마스터와 노드로 이루어진다. 마스터는 노드들을 관리하고 사용자에게 자원을 할당해주는 역할을 하고 노드에서는 자원을 직접적으로 제공하는 역할을 한다. 사용자는 컨테이너 생성에 관련된 이미지 정보라던가 필요한 자원 같은 정보를 YAML 파일로 작성해 마스터에서 실행하게 된다. 그러면 마스터는 해당 요구조건을 충족할 수 있는 노드를 선택해 해당 노드에서 파드를 생성되고 작업이 실행 될 수 있도록 해준다. 이러한 시스템에서 사용자의 편의를 높이고자 콘솔 작업을 통해 YAML 파일을 작성할 필요 없이 웹상에서 인터넷 창을 통하여 간단하게 파드를 생성하는 작업을 할 수 있도록 웹서버를 만들어 연동되도록 하였다. 또한 테라 바이트 이상의 용량이상을 요구하는 대용량 작업을 실행하는 파드도 생성 가능하도록 하기 위해서 대용량 스토리지를 지원해주는 Ceph 을 추가하였다.[4] 이렇게 구성된 시스템의 전체적인 모습은 아래의 그림 1 과 같다.

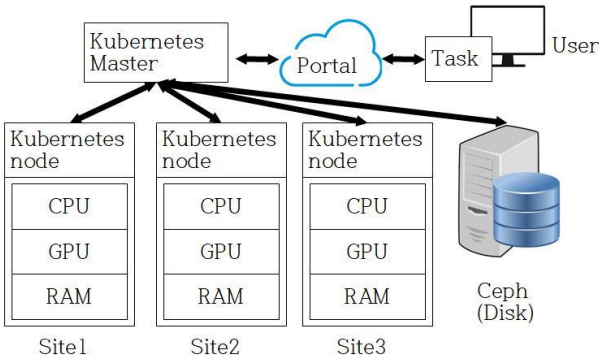


그림 1.시스템 모델
Fig 1.System Model

현재 과학 데이터의 초고속 전송을 위한 네트워크 인프라 구축을 위해 정부 출연 연구기관들을 중심으로 프로젝트를 진행하고 있다. 각 연구기관에서는 고속으로 데이터 전송이 가능한 서버인 DTN(Data Transfer Node)를 설치하여 전송에 사용하고 있다.[5-6] 이러한 용도로 사용되는 서버들을 단순히 전송 용도로만 쓰기 보다는 컴퓨팅 자원을 모아 머신러닝 같이 고성능의 컴퓨팅 또는 계산이 요구되는 작업도 실행이 가능하도록 지원하기 위해 DTN 들을 앞에 언급한 클라우드 시스템의 노드들로 등록해 사용되고 있다.[7]

2.2 ScienceDMZ

과학 데이터의 초고속을 위한 네트워크 인프라 구축 프로젝트에서 있어서 DTN 의 설치 이후 핵심적인 내용으로 ScienceDMZ 가 있다. ScienceDMZ 는 방화벽 우회, 망분리, 네트워크 튜닝 같은 기술들을 적용하여 대용량 과학 데이터의 초고속 전송이 가능한 네트워크를 말한다.[8]

연구기관들에 분산되어 배치되어 있는 DTN 이 사용하는 네트워크에 ACL(Access Control List)설정을 통해서 신뢰성 있다고 판단되는 노드의 통신만 가능하도록 설정하여 방화벽을 우회하여도 문제가 없도록 하였다. 방화벽은 1Gbps 의 전송 속도를 사용하는 일반 네트워크에 최적화되어 있기 때문에 고대역폭 네트워크에서는 성능을 낮추는 요소가 된다. 연구기관의 DTN 들로 구성한 ScienceDMZ 는 그림 2 와 같다. ACL 적용과 방화벽을 우회를 통해 DTN 간의 속도 저하 없이 데이터의 전송이 가능한 환경이 구성된다.

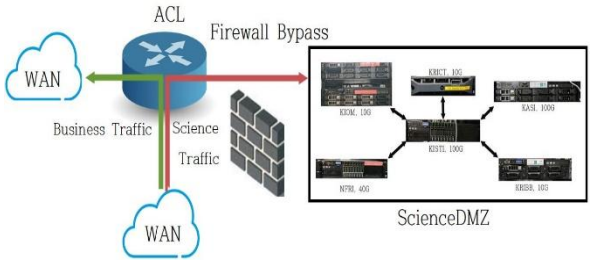


Fig 2.ScienceDMZ 네트워크 구성도

Fig 2.ScienceDMZ Network diagram

네트워크 구성 후 DTN 튜닝을 통해서 네트워크에서 지원하는 대역폭을 최대한 활용할 수 있다. 40~100Gbps 의 네트워크 대역폭 기준으로 MTU(Maximum Transmission Unit)의 사이즈를 기본값인 1500 에서 9000 으로 수정하게 되면 속도가 크게 향상된다. MTU 는 패킷의 최대 크기를 의미한다. 이 값이 커지게 되면 데이터가 여러 개의 패킷으로 분할 전송될 때 보다 줄어든 횟수로 전송하면 되기 때문에 속도가 증가하게 된다. 하지만 MTU 값을 너무 크게 줘서 네트워크 경로에서 감당하지 못하게 되면 다시 분할해서 전송해야 하기 때문에 오히려 속도가 낮아지게 되기에 성능을 최대로 내기 위해서는 적정 값을 찾아 적용해야 한다. MTU 이외에도 CPU 의 성능을 높여주는 설정과 메모리 버퍼 설정 같은 튜닝 방법을 통해 전송 성능을 높일 수 있다.

2.3 컨테이너 기반 클라우드내 ScienceDMZ 적용

연구기관들에 배치되어 있는 DTN 들로 컨테이너 기반 클라우드인 쿠버네티스를 활용해 대규모의 계산 작업을 위해 필요한 자원의 대여가 가능한 클러스터를 구성한다. 이 클라우드 시스템을 통해서 사용자가 머신러닝 작업을 생성해 실행할 수 있는데 이 경우 높은 정확도를 얻기 위해 대규모의 학습 데이터를 가져와 저장하여 학습에 활용하고자 데이터가 저장되어 있는 다른 서버로부터 데이터를 전송 받는 작업을 진행하게 된다. 이 때 대용량 데이터를 클라우드 기본 네트워크를 통해 전송하는 작업은 오버헤드가 크기 때문에 전송시간이 오래 걸려 효율적이지 못한 면이 있다 이러한 점을 해소하기 위해 고대역폭 전송망 구현 기술인 ScienceDMZ 를 클라우드에 적합하게 조정하여 적용해 보았다. 조정된 옵션으로는 MTU 사이즈와 CPU 의 작업을 네트워크 카드에서 처리하도록 하는 Offload 기능이 있어서 이러한 값의 조정을 통해 전송 속도를 향상시킬 수 있다.

Ⅲ. 구현 및 시험

이번 장에서는 DTN 들로 구성된 클라우드에서 ScienceDMZ 기술을 적용해 데이터 전송 테스트를 진행한다. 테스트에 사용된 서버의 성능은 아래의 그림 3 과 같다.

Classification	Kubernetes Master	Kubernetes Node
Product name	PowerEdge R720	PowerEdge R730
CPU	Intel Xeon E5-2680@2.70GHz	Intel Xeon E5-2640@2.40GHz
Memory	16 GB PC4-17000	128 GB PC4-17000
Storage	600 GB	6000 GB
Network Devices	Mellanox 40 GbE	Mellanox 100 GbE
Kernel	Red Hat 5.0.13-1	Red Hat 5.0.13-1

Fig 3.쿠버네티스 서버 하드웨어 스펙

Fig 3.Kubernetes Server Hardware Specification

쿠버네티스의 노드로 사용되고 있는 동일한 성능의 R730 두 대에 쿠버네티스 마스터를 통해 Esnet 에서 제공하고 있는 데이터 전송 소프트웨어 이미지를 사용해 파드를 생성한다. 파드가 물리적으로 서로 다른 노드에 생성되었기 때문에 파드간 통신을 하려면 쿠버네티스에서 서비스를 생성해 주어야 한다. 서비스로 노드 자체의 아이피로 통신할 수 있게 externalIP 로 노드의 IP 를 사용하도록 설정하고 여기에 데이터 전송 소프트웨어에서 사용할 포트들을 정의해주면 된다.

현재의 쿠버네티스에서 서비스를 통해 사용할 포트에 대해서 범위로 설정하는 기능은 지원하지 않기에 일일이 사용할 포트 목록을 작성해주어야 한다. 때문에 클라우드를 통하지 않고 서버간 직접 네트워크 대역폭을 테스트할 때 성능이 잘 나오는 Iperf 대신에 Iperf3, 3.7 버전을 사용하였다. Iperf 에서 네트워크 대역폭 테스트시 클라이언트의 사용 포트가 범위로 설정되고 고정된 포트로는 사용할 수 없으나 Iperf3, 3.7 버전에서는 cport 옵션이 추가되어 고정된 포트로 사용할 수 있게 되었다. 실험에 쓰인 R730 서버를 기준으로 Iperf 사용시 parallel 옵션을 4 정도 설정하고 네트워크 대역폭을 테스트하면 98 Gbps 정도로 네트워크 인터페이스 카드의 성능을 거의 가져와 쓸 수 있지만 Iperf3 에서는 20 Gbps 의 성능정도 밖에 사용하지 못한다. 이러한 성능차이의 주요이유는 두 프로그램 간의 작동방식이 상이하기 때문에 똑같이 parallel 옵션을 주어도 Iperf 에서는 CPU 를 다중 코어를 사용하는 반면 Iperf3 에서는 싱글 코어로만 동작하기 때문에 발생한다.

쿠버네티스에서 생성된 파드에서 Iperf3 를 기본적인 옵션으로만 진행하게 되면 약 2Gbps 의 속도로 전송되는데 이는 연결된 네트워크 대역폭인 100Gbps 대비 약 2%

만 사용하기에 매우 낮은 성능이다. 이러한 문제를 해결하여 네트워크의 성능을 보다 올리기 위해 ScienceDMZ 에서 DTN 튜닝에 사용되는 기술을 가져와 적용해보는 실험을 진행한다. 2 장에서 언급된 DTN 튜닝 방법에서 가장 큰 성능 변화를 보이는 MTU 사이즈 조절을 진행할 수 있다. 기본적으로 설정되어 있는 MTU 값은 1500 이며 이 값을 ScienceDMZ 에서 구성에 사용하는 옵션인 9000 으로 바꾸게 되면 파드에서 나올 때 20 바이트가 붙어 패킷의 크기가 9020 바이트가 되어버려 오히려 성능이 감소되기 때문에 8980 으로 설정해주었다. MTU 사이즈의 조절만으로도 기본 설정의 속도 2Gbps 대비 4 배의 가까운 8Gbps 의 성능을 보여주었다. MTU 사이즈 변경이외에 네트워크 속도에 큰 영향을 준 옵션으로는 오프로드가 있다. 오프로드에는 TCP 체크섬 오프로드(TCP Checksum Offload)와 TCP 세분화 오프로드(TCP Segmentation Offload)가 있는데 각 오프로드에 대해서 설명하자면 다음과 같다. TCP 체크섬 오프로드는 OS 에서 수행하는 체크섬 기능을 네트워크 카드가 대신하도록 하여 서버의 성능을 개선시켜준다 마찬가지로 TCP 세분화 오프로드로 데이터 전송을 위해 패킷 단위로 나누고 다시 합치는 작업들을 네트워크 카드에서 수행하도록 해줌으로써 데이터 전송시 CPU 의 부담이 줄어들어 성능이 향상되는 결과를 보였다. MTU 가 1500 인 상태에서 offload 기능을 사용하는 것만으로도 MTU 튜닝을 했던 것과 비슷하게 4 배 정도의 성능향상을 보였다. 여기에 MTU 값도 8980 으로 설정해주게 되면 15.5 Gbps 의 성능을 얻을 수 있다. 테스트로 얻은 결과 데이터들을 아래의 그림 4 를 통해 차트로 정리하였다.

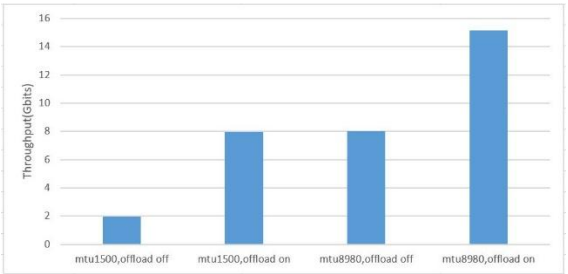


Fig 4.클라우드내에서 ScienceDMZ 옵션 적용 비교
Fig 4.Comparison of applying ScienceDMZ option in cloud

Ⅳ. 결론

본 논문에서는 출연연 중심의 연구망 구축 프로젝트를 통해 고대역폭 데이터 전송망의 DTN 으로 사용되고 있는 노드들의 여유자원을 머신러닝 작업이나 대규모의 과학 데이터 계산등에 활용하기 위한 방안으로 쿠버네티스를 기반으로 통합 자원 관리 클러스터 구현해 사용하고 있다. 이러한 클러스터 내에서 데이터 전송 속도를 향상시키기 위해 ScienceDMZ 에서 사용되는 방법들을 가져와 클라우드에 맞게 변경 적용하여 기본 설정 대비 7.7 배 높은 네트워크 전송 속도를 얻을 수 있었다.

참 고 문 헌

[1] Y. M. Bae, S. J. Jung, and W. Y. Soh, “Comparative Analysis of the Virtual Machine and Containers Methods through the Web Server Configuration,” *JKIICE*, vol. 18, no. 11, pp. 2670-2677, 2014.

[2] S. G. Lee and W. J. Seok, “Construction of Highly Scalable Cloud Computing to Run Deep Learning Framework,” in *Proc. The Institute of Electronics and Information Engineers*, pp. 147-149, 2017.

[3] What is Kubernetes, Retrieved 2019 [Online], from: https://kubernetes.io

[4] Introduction to Ceph, Retrieved 2019 [Online], from: https://ceph.com/ceph-storage

[5] W. J. Seok, W. T. Hong, J. S. Kwak, J. H. Moon, “DTN-aware Store-and-Forward Data Transfer for Exabyte Scale Science Big-data,” *J-KICS*, vol. 4 2, no. 10, pp. 1991-1998, 2017.

[6] E. Dart, L. Rotman, B. Tierney, H. Hester and J. Zuraski, “The ScienceDMZ: A Network Design Pattern for Data-Intensive Science,” *Scientific Programming*, vol. 22, no. 2, pp. 173-185, 2014.

[7] I. Altintas, K. Marcus, I. Nealey, S. L. Sellars, J. Graham, D. Mishin, J. Polizzi, D. Crawl, T. DeFanti, and L. Smarr, “Workflow-driven distributed machine learning in CHASE-CI: A cognitive hardware and software ecosystem community infrastructure,” *IPDPSW*, pp. 865-873, 2019.

[8] L. Smarr, C. Crittenden, T. DeFani, J. Graham, D. Mishin, R. Moore, P. Papadopoulos and F. Wurthwein, “The Pacific Research Platform: Making High-Speed Networking a Reality for the Scientist,” in *Proc. ACM*, pp. 29, 2018.

멀티 서비스 체이닝 환경에서 VNF 자원 수요 예측 기법

조윤영, 장석원, 양혜임, 백상헌*
고려대학교

yunyoung0309@gmail.com, imsoboy2@gmail.com, yanghyeimm@gmail.com,
*shpack@gmail.com

VNF Resource Demand Prediction Method in Multi Service Chaining Environment

Yunyoung Cho, Seokwon Jang, Hyeim Yang, Sangheon Pack*
Korea Univ.

요 약

네트워크 기능 가상화 (NFV: Network Function Virtualization)는 기존 네트워크 대비 유연하고 기민한 환경을 제공하지만, 수동적으로 네트워크를 관리해야 한다는 문제점이 존재한다. 최근에는 이를 해결하기 위해서 머신러닝 기술을 네트워크 관리에 활용하는 연구가 활발하게 진행되고 있다. 본 논문은 멀티 서비스 체이닝 환경에서 가상 네트워크 기능 (VNF: Virtualized Network Function)의 미래 자원 수요를 예측하는 새로운 모델을 제안한다.

Ⅰ. 서 론

네트워크 기능 가상화 (NFV: Network Function Virtualization)는 기존의 전통적인 네트워크 환경에서 전용 하드웨어 장치를 소프트웨어로 구현한 기술로 NFV 환경은 네트워크 사업자 또는 운영자의 기존 네트워크 장비 대비 장비 투자비 (CAPEX)와 운용비용 (OPEX)을 절감한다. 또한 네트워크 관리자의 요구사항에 따라 동적으로 네트워크 구성을 변경하여 운용할 수 있는 유연성을 제공한다.

NFV 환경의 중요한 개념 중 하나는 서비스 기능 체이닝 (SFC: Service Function Chaining)이다. SFC 는 다수의 가상 네트워크 기능 (VNF: Virtual Network Function)들을 하나의 논리적인 체인으로 구성한 뒤 순차적으로 처리하는 기술이다. 그림 1 은 3 개의 SFC 가 존재하는 멀티 서비스 체이닝 환경이고, VNF1, VNF2, VNF3 와 같이 한 개의 VNF 에 여러 개의 서비스 기능 체인이 지나갈 수 있다. SFC 는 NFV 와 함께 SDN (SDN: Software Defined Networking)과 결합하여 유연한 네트워크 제어 및 관리를 가능하게 한다.

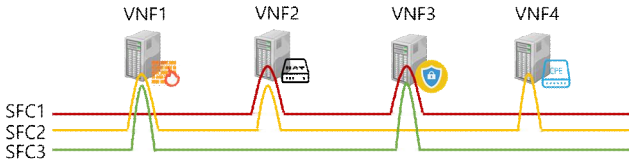


그림 1. 멀티 체이닝 환경

하지만 동적인 네트워크로의 변화에도 불구하고, 가상 네트워크 생성 및 관리에는 여전히 관리자가 개입해야 한다는 문제가 있다. 최근에는 수동적인 관리 방식의 문제점을 해결하기 위해 머신러닝 (Machine-Learning) 기술을 네트워크 관리에 적용하는 방법이 관심을 가지게 되었다. [1]

네트워크 관리에 있어서 핵심적인 요구사항은 네트워크 자원을 최적으로 관리하는 것이다. NFV 환경은 여러 VNF 들이 순차적으로 연결된 SFC 형태로 네트워크 서비스를 제공하기 때문에 네트워크 가상화 환경에서 최적화된 자원관리 문제는 SFC 를 구성하는 각각의 VNF 들의 자원 수요를 정확하게 예측하는 문제로 귀결된다.

본 논문에서는 머신러닝을 이용해서 멀티 서비스 체이닝 환경에서 VNF 의 자원 수요를 예측하는 기법을 제안한다. 이 후 본 논문의 결론을 내리고 향후 연구 과제에 대해 논의한다.

Ⅱ. 관련 연구

최근에 머신러닝을 적용한 네트워크 관리 기법에 대한 연구들이 활발하게 진행되고 있다. 논문 [2]에서는 머신 러닝을 활용하여 VNF 의 스케일링 문제를 결정하는 방법을 제안하며, 논문 [3]에서는 신경망 모델 중 하나인 Long Short Term Memory (LSTM) 모델을 이용해서 VNF 의 미래 자원 수요를 예측한다.

논문 [4]에서는 본 논문에서 제안하는 방법과 유사하게 SFC 를 고려한 VNF 의 자원 수요를 예측하는 기법을 제안한다. 저자는 Graph Neural Network (GNN)을 사용하여 각 VNF 에 직접적으로 연결된 VNF 의 데이터를 기반으로 미래의 자원 수요를 예측하고 있다. 하지만 GNN 은 시계열 데이터의 특성 전체를 사용할 수 없고 VNF 에 여러 체인이 지나가는 환경을 고려하고 있지 않다는 한계점이 존재한다.

논문 [5]의 저자는 SFC 를 구성하는 VNF 의 자원 정보를 사용해서 각 VNF 의 미래 자원 수요를 예측한다. 해당 논문에서는 Target Dependent LSTM 모델에 머신러닝 기술인 Embedding 과 Attention 을 결합하여 자원 수요 예측 정확도를 높이고 있다. 하지만 해당 모델도 멀티 서비스 체이닝 환경을 고려하지 않고 있다는 한계점이 존재한다.

Ⅲ. 멀티 서비스 체이닝 환경을 고려한 기법 제안

본 논문에서 제안하는 모델은 그림 2 와 같이 동작하며, 3 개의 LSTM 으로 구성된다. LSTM 을 사용한 이유는 LSTM 은 CPU 사용량과 같이 시계열 데이터에 학습에 좋은 성능을 보이는 모델이기 때문이다.

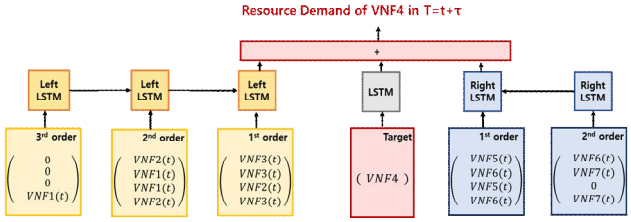


그림 2. 제안하는 모델 구조

머신러닝 모델은 입력 데이터로 개별 VNF 의 자원 정보와 SFC 를 구성하는 모든 VNF 의 자원 정보를 사용한다. 표 1 은 4 개의 SFC 를 구성하는 모든 VNF 들에 대한 자원 정보 데이터를 나타낸다. 하지만 실제 모델에 입력되는 데이터는 표 2 에 제시된 것과 같이 예측하고자 하는 VNF 를 기준으로 VNF 들에게 순서를 부여하는 전처리 과정이 필요하다. 전처리 과정은 서비스 기능 체인 안에서 타겟 VNF 와 가장 인접한 VNF 들에게는 가장 낮은 번호를 부여하고 가장 멀리 위치한 VNF 들에 대해서는 가장 높은 번호를 부여하는 과정을 수행한다.

SFC 1	VNF2, VNF3, **VNF4**, VNF5, VNF6
SFC 2	VNF1, VNF3, **VNF4**, VNF6, VNF7
SFC 3	VNF1, VNF2, **VNF4**, VNF5
SFC 4	VNF1, VNF2, VNF3, **VNF4**, VNF6, VNF7

표 1. SFC 를 구성하는 VNF 들에 대한 자원 정보

Target	왼쪽 LSTM 입력값			오른쪽 LSTM 입력값	
	1st order	2nd order	3rd order	1st order	2nd order
VNF4	VNF3	VNF2		VNF5	VNF6
	VNF3	VNF1		VNF6	VNF7
	VNF2	VNF1		VNF5	
	VNF3	VNF2	VNF1	VNF6	VNF7

표 2. 전처리한 데이터 형태

전처리 과정 후에는 왼쪽 LSTM 의 입력으로 타겟 VNF 를 기준으로 왼쪽에 위치한 VNF 들에 대한 자원 정보를 입력하며 가장 낮은 번호를 부여받은 VNF 들을 가장 나중에 입력한다. 이런 입력방식을 선택한 이유는 가장 가까운 VNF 들의 자원 정보가 타겟 VNF 의 자원 수요를 예측하는데 있어 가장 큰 영향을 끼치게 하기 위해서이다. 오른쪽 LSTM 도 왼쪽과 마찬가지로 가장 낮은 번호를 부여받은 VNF 들에 대한 정보가 가장 나중에 입력된다.

하지만 해당 방법은 VNF 에 여러 서비스 기능 체인이 지나가는 경우 입력 데이터에 중복적인 데이터가 많다는 문제점이 있다. 이를 해결하기 위해서 Auto Encoder [6]를 이용해서 입력 값의 중복적인 데이터를 줄일 수 있을 것으로 보고 있다.

Ⅳ. 결 론

본 논문에서는 멀티 서비스 체이닝 환경에서 VNF 의 자원 수요를 예측하는 새로운 모델을 제안한다. 향후에는 해당 모델 검증과 SFC 시나리오를 구현한 실험 환경을 구축하여 모델 검증을 수행할 예정이다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학 ICT 연구센터지원사업의 연구결과로 수행되었음 (IITP-2019-2017-0-01633)

이 논문은 2017 년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No. 2017-0-00195, 멀티 서비스를 지원하는 프로그래머블 스위치 제어 기술 개발)

참 고 문 헌

[1] S. Ayoubi, N. Limam, M. A. Salahuddin, et al. Machine learning for cognitive network management. IEEE Communications Magazine, 2018.

[2] S. Rahman, T. Ahmed, M. Huynh, et al. “Auto-scaling vnfs using machine learning to improve qos and reduce cost,” in *Proc. IEEE International Conference on Communications (ICC) 2018,* Kansas City, US, May 2018.

[3] S. Gupta and D. A. Dinesh. “Resource usage prediction of cloud workloads using deep bidirectional long short term memory networks,” in Proc. IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS) 2017, Odisha, India, December 2017.

[4] R. Mijumbi, S. Hasija, S. Davy, A. Davy, B. Jennings, R. Boutaba, "A connectionist approach to dynamic resource management for virtualised network functions", 2016 12th International Conference on Network and Service Management (CNSM), pp. 1-9, Oct 2016.

[5] Hee-Gon Kim, Do-Young Lee, Se-Yeon Jeong, Heeyoul Choi, Jae-Hyung Yoo and James Won-Ki Hong, "Machine Learning-Based Method for Prediction of Virtual Network Function Resource Demands," in Poc. IEEE Conference on Network Softwarization (NetSoft) 2019, Paris, June 2019.

[6] P. Baldi, "Autoencoders unsupervised learning and deep architectures", Proc. ICUTLW, pp. 37-50, 2012.

엣지 컴퓨팅을 위한 컨테이너 네트워크 성능 비교에 관한 연구

윤정한†, 이건*, 신상호*
†한국과학기술원 전기및전자공학부
*SK 텔레콤 Cloud Labs

†cero512@kaist.ac.kr, *gunine@sk.com, *sanghoshin@sk.com

A study on assessing container network performance for edge computing

Junghan Yoon†, Jian Li*, Sangho Shin*
†Department of Electrical Engineering, KAIST
*Cloud Labs, SK Telecom

요 약

클라우드 기술이 발전됨에 따라 경량화된 컨테이너 기술에 관심이 집중되고 있다. 컨테이너 네트워킹은 높은 수준의 클라우드 서비스를 제공하는데 있어 핵심 기술로 Container Network Model (CNM)과 Container Network Interface (CNI)와 같은 다양한 구현 모델이 존재한다. 그 중 CNI 는 다양한 컨테이너 플랫폼에서 채택한 de-facto standard 로 CNI 네트워킹 모델을 구현한 여러 플러그인들이 존재하며 구현 방식이 다름에 따라 성능면에서 큰 차이를 보인다. MEC (Multi-Access Edge Computing)은 이동통신서비스를 이용하려는 사용자와 가장 가까운 곳에 서버를 위치시켜 서비스를 제공하는 기술로 multi-tenancy 제공을 위하여 Container-on-VM (CoV) 형상으로 구축한다. 본 논문에서는 CoV 형상에 다양한 CNI 네트워크 플러그인들을 적용하여 성능을 측정하였고 이를 통하여 네트워크 성능 저하를 발생시키는 원인을 분석하였다. 분석 결과는 향후 저지연을 위한 CoV 형상을 구축함에 있어 좋은 reference 로 사용될 수 있을 것으로 판단된다.

Ⅰ. 서론

MEC 는 이동통신서비스를 사용자 단말과 가까이에 위치시켜 대용량 및 초저지연 서비스를 제공하는 클라우드 컴퓨팅 기술이다. 인터넷을 통하여 제공되는 퍼블릭 클라우드 서비스와는 달리 MEC 서비스는 이동통신사업자 망 내에 클라우드 플랫폼을 위치시켜 제공하기 때문에 지연에 민감한 AR(Augmented Reality), VR(Virtual Reality), 스마트 팩토리 및 자율주행 등 다양한 차세대 서비스들을 실현하는데 있어 핵심 기술로 간주되고 있다.

컨테이너 기술은 경량화된 애플리케이션 격리 기술로 차세대 클라우드 서비스 실현에 있어 핵심 요소 기술이다. Kubernetes 는 컨테이너 orchestration 플랫폼 중 하나로 구글이 자사 서비스 제공을 위하여 개발하였던 Borg 를 기반으로 한 오픈소스 프로젝트이다. Kubernetes 는 컨테이너를 관리하는 de-facto standard 플랫폼으로 일반 IT 서비스부터 최근에는 이동통신 서비스에도 적용이 검토되는 등 다양한 형태의 서비스를 제공하는 목적으로 사용되고 있으며, MEC 서비스 또한 Kubernetes 를 통하여 제공될 것으로 전망되고 있다. Kubernetes 는 경량화, 효율성 측면에서 가상 머신 기술 대비 우수하나 컨테이너의 태생적인 문제인 보안 취약점을 그대로 가지고 있어 높은 수준의 자원 격리가 힘들며 특히 시스템 커널을 공유하고 있다는 점에서 multi-tenancy 목적으로 사용하기가 힘들다. MEC 플랫폼에는 이동통신사 서비스들뿐만 아니라 일반 IT 개발사들에서 개발한 서비스들도 같이 호스팅되며, MEC 서비스의 QoS 를 보장하기 위하여 각 MEC 서비스들은 독립된 자원을 할당받아 사용하여야 한다. 이런 특성은 자원 효율성을 극대화한 컨테이너 기술과는 방향성 측면에서 상충되는 바 컨테이너 기술을 MEC 에 활용하기 위하여 일련의 최적화가 필요하다.

이 문제를 해결하기 위하여 다양한 연구들[1][2] 이 진행되어 왔으며 그 중 자원의 격리성과 자원의 효율성 두 가지 장점을 모두 보유한 Container-on-VM (CoV) 방법이 구현 난이도 및 기술 성숙도 등의 측면에서 인정받아 여러 Cloud Service Provider (CSP)들이 이미 지원하고 있다. CoV 형상에서 컨테이너들은 가상 머신 내에 위치하여 가상 머신에서 제공하는 커널 자원을 공유하고 있기 때문에 가상 머신 수준의 보안성을 제공하지만, 컨테이너 간 통신을 위하여 VM 네트워킹 스택과 컨테이너 네트워킹 스택을 이중으로 거쳐야하기 때문에 네트워크 성능 열화가 발생한다. 이는 지연에 상대적으로 덜 민감한 퍼블릭 클라우드 서비스 제공 시에는 문제가 되지 않으나 AR 과 VR 와 같이 지연에 민감한 MEC 서비스를 제공하는 이동통신사업자 경우 상당히 치명적이다. Kubernetes 는 CNM, CNI 두 모델 중 CNI 네트워킹 모델[3]을 채택하여 사용하고 있다. CNI 는 컨테이너 네트워크의 동작에 필요한 기능 요구사항만을 정의하고 있기 때문에 CNI 구현에 사용된 기술이 다름에 따라 네트워크 성능 또한 큰 차이를 보이게 된다.

본 논문에서는 CoV 의 네트워크 성능에 영향을 주는 다양한 VM 네트워킹 스택과 CNI 플러그인의 조합(combination)을 만들고 각 조합에서의 네트워크의 처리율(throughput)과 지연(latency)의 측정을 통하여 MEC 유스케이스에 가장 적합한 조합을 찾아내고 해당 측정 결과를 보이게 된 원인을 분석 및 도출하고자 한다. 본 논문을 통하여 도출된 결과는 향후 CoV 기반의 MEC 인프라 구축에 있어 좋은 reference 로 활용될 것으로 전망된다.

Ⅱ. 관련 연구

기존 CNI 성능 평가 연구[4][5]들에서 다양한 CNI plugin 의 성능을 측정, 비교 및 분석했지만 CoV 형상에

초점을 맞추어 진행한 연구가 아닌 관계로 MEC 에 실험결과를 활용하기 힘들다. 또한 host-device CNI 에 대한 성능 검증이 고려되지 않았다. CoV 형상을 고려한 연구[6] 또한 성능의 지표로 최대 처리율을 우선적으로 고려하였기에 본 연구에서는 MEC 를 위한 최소 지연에 초점을 둔 새로운 성능 평가를 하게 되었다.

Ⅲ. 실험방법 및 실험결과

VM 네트워크의 VxLAN[7] 사용여부와 컨테이너 네트워크의 pass-through 여부에 따라 다음과 같이 4 가지 네트워크 모델을 구성했다.

표 1 CoV 네트워크

CoV 네트워크	PT-flat	PT-vxlan	calico-flat	calico-vxlan
VM	Flat	VxLAN	Flat	VxLAN
CNI plugin	host-device	host-device	Calico	Calico

Pass-through 하지 않았을 때 사용하는 기본 CNI 플러그인으로는 Calico[8]를 선정하였다. Calico 는 IP-in-IP 캡슐화 방식을 사용하여 컨테이너 간 통신을 가능하게 하는 CNI 플러그인이다. 완성도가 높고 커뮤니티의 지원도 활발해서 온-프레미스 환경에서 Kubernetes 네트워크를 구축하는 목적으로 가장 많이 사용되는 CNI 플러그인이기 때문에 본 실험에서도 사용하였다. Host-device 는 pass-through 를 실현해주는 CNI 로 해당 플러그인을 사용하면 VM 에 할당된 가상 포트를 컨테이너로 직결시켜준다.

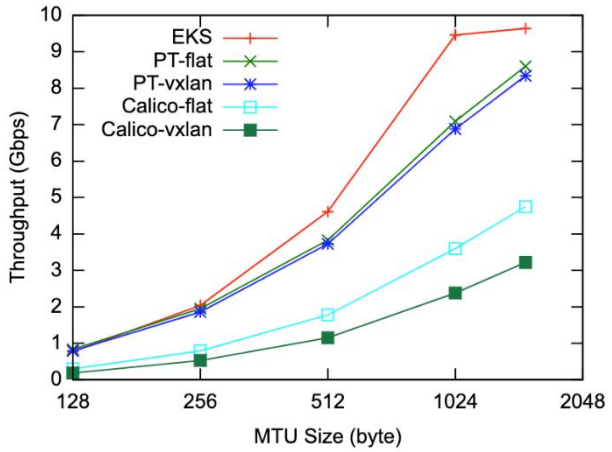


그림 1 iPerf 를 이용한 처리율 측정 결과

퍼블릭 클라우드에서 제공하는 Managed Kubernetes 서비스와의 성능 비교를 위해 AWS-EKS (Elastic Kubernetes Service)에서도 같은 실험을 수행한다. 다만 AWS-EKS 는 물리적인 구성을 변경할 수 없기에 동일선상에서의 성능 평가가 어려움을 감안해야 한다.

Iperf[9] MTU 사이즈를 변수로 두고 8 개의 thread 를 사용하여 120 초간 측정한다. MTU 사이즈가 작아지면 iperf 에서 전송하는 데이터그램이 더 작은 프레임으로 나누어지고 프로세싱도 더 빈번해져서 처리율이 낮아지게 된다. 두 호스트가 대역폭이 10Gbps 인 이더넷 케이블로 연결되어 있으므로 네 조합 모두 처리율이 10Gbps 를 초과할 수 없다.

컨테이너 네트워크를 pass-through 했을 때는 VxLAN 의 유무가 처리율에 주는 영향이 비교적 작지만 Calico 를 사용하는 경우 VxLAN 의 영향이 더 커진다. 이는 네트워크에서의 병목보다는 시스템에서의 병목 때문인 것으로 보인다. 호스트 입장에서는 Calico 와 VxLAN 둘 다 호스트의 하드웨어 자원을 소모하고 있는 상황이다. 이 때문에 calico-vxlan 의 경우, Calico 가 이미 자원을 소모하고 있는 상황에서 VxLAN 이 자원을 더 소모하게 되어서 처리율이 비교적 더 떨어지게 된다.

기본 MTU 사이즈인 1500B 에서의 처리율은 PT-flat 이 8.6Gbps, PT-vxlan 이 8.34Gbps, calico-flat 이 4.75Gbps, calico-vxlan 이 3.22Gbps 이다. 이를 통해 컨테이너 네트워크를 pass-through 할 때 처리율이 80% 향상되며, VxLAN 대신 플랫 네트워크를 사용했을 때 처리율이 3% 향상되는 것을 확인할 수 있다 (그림 1).

AWS EKS (Elastic Kubernetes Service)의 경우 기본 MTU 인 9001 (점보 프레임)를 쓸 경우 정확하게 10Gbps 로 측정되고 순간적으로는 10Gbps 를 넘기도 한다. 이는 AWS 에서 호스트간 통신으로 대역폭이 10Gbps 를 초과하는 이더넷 케이블을 사용하고 있고, VM 인스턴스를 할당하면서 VM 간 처리율을 소비자 수준 협약(SLA)에 의해 10Gbps 로 제한했기 때문으로 보인다. 본 실험에서는 EKS 와 물리적인 구성을 일치시킬 수 없기 때문에 테스트베드에서 구성한 모델과 EKS 의 처리율을 직접적으로 비교하기 힘들다. 하지만 그럼에도 불구하고 후술할 sockperf[10] 실험에서 측정하는 지연 성능의 경우 테스트베드와 EKS 의 차이가 크지 않은 것을 확인할 수 있다.

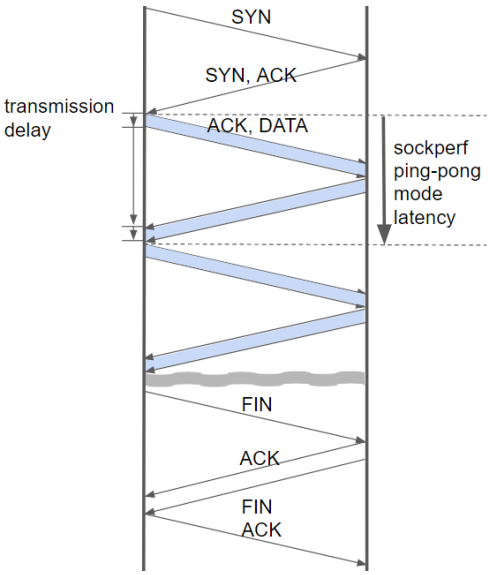


그림 2 sockperf ping-pong 모드 지연 측정 방법

Sockperf 실험은 변수의 개입을 최대한 배제하기 위해 120 초 동안 실행하고 지연의 최소값을 측정했다. 그림 2 와 같이 sockperf ping-pong 모드 실험은 하나의 쓰레드로 MTU 사이즈보다 작은 사이즈의 프레임을 주고받기 때문에 큐잉 (queueing)이 발생하지 않는다. 또한 해당 테스트베드에서 호스트 간 거리가 매우 가까워 전파 지연은 무시 가능하다. (그림 2)

Sockperf 컨테이너 밖으로 전송되는 프레임의 사이즈는 sockperf 에서 전송하는 데 사용하는 메시지의 사이즈에 따라 다르지만 옵션을 주지 않을 경우 80B 이다. 이는 sockperf 의 기본 메시지 사이즈인 14B 에 타임스탬프 옵션이 추가된 TCP 헤더(32B), IP 헤더(20B), 이더넷 헤더(14B)를 더한 값이다. 이후 CNI-플러그인을 calico 를 사용할 경우 IP 캡슐화를 위한 20B 의 헤더가 추가되고, VM 네트워크로 VxLAN 을 사용하는 경우 50B 의 헤더가 추가된다. 둘 다 사용될 경우 70B 의 헤더가 추가된다.

전송 지연은 호스트 프레임의 사이즈에 따라 다르다. PT-flat 의 경우 컨테이너의 인터페이스가 호스트의 인터페이스이므로 프레임 사이즈가 80B 로 메시지 사이즈와 같다. PT-vxlan 의 프레임 사이즈는 50B 의 VxLAN 헤더가 추가되어 130B 가 된다. calico-flat 의

프레임 사이즈는 20B 의 calico 헤더가 추가되어 100B 가 되고, calico-vxlan 의 프레임 사이즈는 70B 가 추가되어 150B 로 증가한다. 대역폭이 10Gbps 인 이더넷 케이블을 통해 전송되고 sockperf 클라이언트와 서버, 총 두 번의 전송 지연이 발생하므로 PT-flat 의 전송 지연은 128ns, PT-vxlan 의 전송 지연은 208ns, calico-flat 의 전송 지연은 160ns, calico-vxlan 의 전송 지연은 240ns 가 된다. 따라서 sockperf ping-pong 모드로 측정한 지연에서 각 모델의 전송 지연 값을 제외하면 왕복 1 회에 소모되는 처리 지연을 알 수 있다. Sockperf 최소 지연 측정 결과는 다음과 같다.

표 2 Sockperf ping-pong 모드 지연 측정 결과

CoV 네트워크	EKS	PT-flat	PT-vxlan	calico-flat	calico-vxlan
지연[us]	21.724	21.468	26.806	29.697	35.621

VM 커널 내의 네트워크 스택과 호스트 커널 내의 네트워크 스택을 거쳐 여러 처리 지연이 누적되었지만 PT-flat 의 처리 지연과 calico-flat 의 처리 지연의 차이를 통해 calico 의 처리 지연을 알 수 있다. 둘의 차이는 8.229us (= 29.697us - 21.468us)이며, 전송 지연이 calico-flat 이 32ns 더 크기 때문에 처리 지연의 차이는 8.197us 가 된다. Calico 는 전송 및 수신 시 각각 한번씩 처리하므로 차이의 절반인 4.099us 가 calico 의 처리 지연 값이 된다. 동일한 방법으로 PT-vxlan 과 PT-flat 의 차이를 통해 VxLAN 의 처리 지연 시간을 구하면, 2.629us (= (26.806us - 21.468us - 80ns) / 2)를 얻을 수 있다. 따라서 calico 의 처리 지연이 VxLAN 의 처리 지연보다 약 50% 더 크다. (4.099 / 2.629 = 1.559)

EKS 의 경우 CNI-플러그인으로 ENI (Elastic Network Interface)를 사용하기 때문에 calico 가 없는 것은 알 수 있지만 VxLAN 의 사용 여부와 헤더 사이즈를 알 수 없기 때문에 각각의 처리 지연을 계산할 수 없다. 다만 전체 지연 시간이 PT-flat 과 PT-vxlan 의 사이인 것으로 봐서 CNI-플러그인으로 calico 를 그대로 쓰는 것보다는 컨테이너 네트워크를 pass-through 하는 것이 더 경쟁력이 있음을 확인할 수 있다.

네트워크의 부하가 적은 환경에서 컨테이너 네트워크의 pass-through 여부에 따라 지연 시간에 최소 4.099us 의 차이가 발생하는 것을 확인했으므로 점차 부하를 늘리며 이 차이가 얼마나 영향을 미치는지도 확인하였다. Iperf 실험에서와 마찬가지로 MTU 사이즈를 줄여서 부하를 늘릴 수 있으나 sockperf 실험에서는 전송하는 메시지의 크기를 증가시킴으로써 부하를 늘렸다. Iperf 는 기본 메시지 사이즈가 128KB 이므로 MTU 사이즈를 줄일 때 큰 효과가 나타나는 반면, sockperf 의 기본 프레임 사이즈는 80B 이므로 MTU 사이즈를 줄여도 부하가 크게 발생하지 않기 때문이다. 컨테이너의 MTU 사이즈를 80B 미만으로 줄이면 왕복 1 회에 2 개 이상의 프레임을 전송해서 큐잉을 만들 수 있으나 MTU 사이즈는 68B 미만이 될 수 없기 때문에 3 개 이상의 프레임을 만들 수 없다. (RFC 791)

14B 와 1000B 에서의 결과를 보면 알 수 있듯 메시지 사이즈가 MTU 사이즈보다 작은 경우에는 큐잉이 발생하지 않아서 지연이 거의 증가하지 않는다. 그러나 그럼에도 불구하고 모든 모델에서 지연이 1 ~ 2us 증가했는데, 이는 메시지 사이즈의 차이만큼 전송 지연이 증가했기 때문이다. (986B / 10Gbps * 8b/B * 2 = 1.58 us)

메시지 사이즈가 2000B 가 되면 MTU 사이즈를 초과한다. 이로 인해 큐잉 지연이 발생하고 전송지연도 더 커져서 전체적으로 지연이 크게 증가한다. 작은 사이즈의 메시지를 전송할 때는 유사했던 지연도 메시지의 크기가 증가함에 따라 비교적 큰 차이를 보이게 된다. Iperf 실험에서와 마찬가지로 PT-flat 과 PT-vxlan 의 차이가 크지 않으며 calico 를 포함한 네트워크의 지연은 더 큰 것을 확인할 수 있다. EKS 의 경우 PT-flat 과 유사한 모습을 보인다. (그림 3)

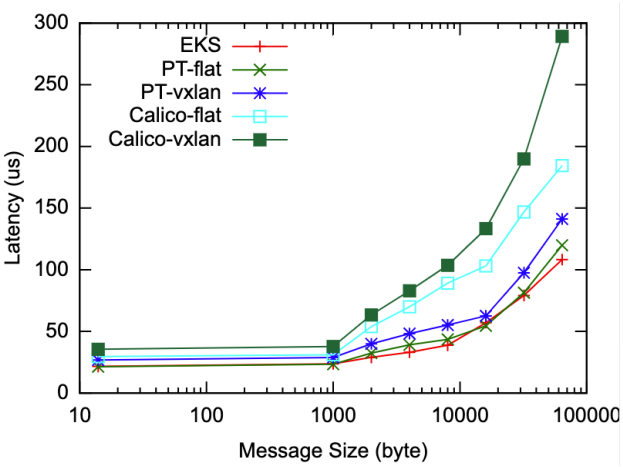


그림 3 sockperf 를 이용한 지연 측정결과

다음 실험으로, 실제 Use Case 에 준하는 수준의 부하를 테스트하기 위해서 많은 소켓을 사용하고 대용량 메시지를 전송할 수 있는 Apache JMeter[11]를 사용했다. Apache JMeter 는 매 요청마다 다른 쓰레드로 접속하기 위해 출발지 포트 번호를 매번 변경한다. 따라서 sockperf 와 달리 TCP 소켓이 요청마다 하나씩 생성되고, TCP 연결과 종료 과정이 매번 포함된다. (그림 4)

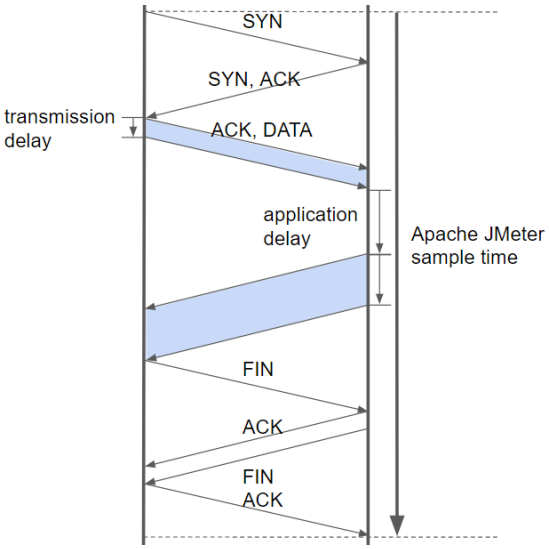


그림 4 Apache JMeter 를 이용한 지연 측정 방법

REST API 통신이 일반적인 어플리케이션에서 가장 보편적으로 사용되는 통신 방법이므로 HTTP 요청만을 사용했고 30 개의 thread 로 120 초간 실험을 수행하였다. 애플리케이션 수준에서는 몇 개의 패킷이 최소의 지연으로 응답하는 것보다는 대다수의 패킷이 특정 값 이내의 지연을 갖도록 보장하는 것이 더 중요하기 때문에 네트워크 지연의 지표로 95 percentile 값을 사용하였다. 95 percentile 값은 95%의 응답이 이 값 이내에 클라이언트에 도착했다는 의미를 가진다. 모바일 트래픽의 HTTP method 중에서는 GET 이 가장 많이 사용되며[12], GET 은 일반적으로 응답 메세지 크기가 요청의 사이즈보다 크다. 따라서, HTTP 요청 메세지 크기는 62B 로 동일하게 유지했고 부하를 조절하기 위해 HTTP 응답 사이즈는 1KB 부터 10MB 까지 증가시키며 측정하였다.

REST API 통신이 일반적인 Application 에서 가장 보편적으로 사용되는 통신 방법이므로 HTTP 요청만을 사용했고 30 개의 thread 로 120 초간 실험을 수행하였다. 어플리케이션 수준에서는 소수의 패킷이 최소의 지연으로 응답하는 것보다는 대다수의 패킷이 특정 값 이내의 지연을 갖도록 보장하는 것이 더 중요하기 때문에 네트워크 지연의 지표로 95th percentile 값을 사용하였다. 95th percentile 값은 95%의 응답이 이 값 이내에 클라이언트에 도착했다는 의미를 가진다. 모바일 트래픽의 HTTP method 중에서는 GET 이 가장 많이 사용되며[12], GET 은 일반적으로 응답 메세지 크기가 요청의 사이즈보다 크다. 따라서, HTTP 요청 메세지 크기는 62B 로 동일하게 유지했고 부하를 조절하기 위해 HTTP 응답 사이즈는 1KB 부터 10MB 까지 증가시키며 측정하였다.

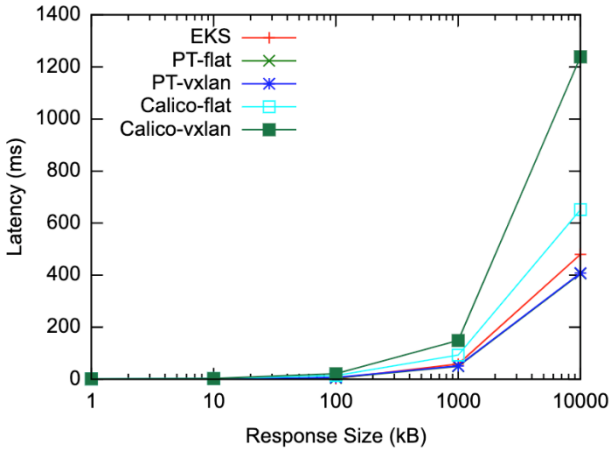


그림 5 Apache JMeter 를 이용한 지연 측정 결과

PT-flat 과 PT-vxlan 의 경우 전구간에서 1 ~ 2ms 밖에 차이가 나지 않아서 그래프가 거의 겹쳐 있는 것을 확인할 수 있다. 또한 부하가 클수록 calico-vxlan 과 다른 모델 간의 차이가 크게 발생한다. 10MB 부하 테스트에서는 PT-flat 의 지연이 406ms, PT-vxlan 의 지연이 408ms, calico-flat 의 지연이 652ms 인 반면, calico-vxlan 의 지연은 1239ms 나 발생한다. 이는 PT-flat 과 PT-vxlan 의 3 배에 달하는 지연이다. (그림 5)

모바일 트래픽의 REST API 에 대한 관련 연구[12]에 따르면 모바일 트래픽의 REST API 응답의 크기는 대부분 1KB ~ 10KB 구간에 분포한다. 1KB 부하 테스트에서는 PT-flat 의 지연은 1ms, PT-vxlan 의 지연은 1ms, calico-flat 의 지연은 2ms, calico-vxlan 의 지연은 2ms 로, 여전히 PT-flat, PT-vxlan 의 지연과 calico-vxlan 의 지연은 2 배의 차이가 발생한다. 10KB 부하 테스트에서도 PT-flat 은 1ms, PT-vxlan 은 2ms, calico-flat 은 3ms, calico-vxlan 은 3ms 가 발생해서 calico-vxlan 의 지연이 PT-flat 보다 3 배 크고 PT-vxlan 보다는 1.5 배 크다.

EKS 의 경우 100KB 부하 테스트까지는 PT-flat 보다 지연이 작지만 1MB 의 부하 테스트에서는 58ms, 10MB 의 부하 테스트에서는 480ms 로 PT-vxlan 보다도 지연이 크다. 이는 EKS 가 더 큰 부하를 갖게 된 것이 원인으로 보인다. 그림 6 은 각 경우의 부하가 얼마나 큰지 확인하기 위해 지연과 동시에 측정한 처리율을 도식화한 그래프이다. 1MB 이상의 부하 테스트에서는 모든 네트워크가 포화되어 처리율이 iperf 로 측정한 최대 처리율에 수렴하는 것을 확인할 수 있다.

처리율 측면에서 VM 네트워크로 Flat 네트워크를 사용하는 경우 VxLAN 대비 3%의 성능 향상이 있었지만 CNI 플러그인으로 pass-through 방식을 사용한 경우 calico 대비 80%의 성능 향상이 있었다. 지연 측면에서 Flat 네트워크를 사용하면 VxLAN 대비 2.629us 의 처리 지연이 감소했고 CNI 플러그인으로 pass-through 방식을 사용할 경우 4.099us 의 처리 지연을 줄일 수 있었다.

이러한 처리율과 처리 지연의 차이가 실제 어플리케이션 수준에서 어느 정도의 성능 차이를 발생시키는지 알아보기 위해 Apache JMeter 테스트도 수행했다. Apache JMeter 실험 결과를 보면, PT-flat 과 PT-vxlan 이 지연 성능면에서 1 ~ 2ms 의 차이만 보이는 반면 calico-flat, calico-vxlan 은 PT-flat 과 PT-vxlan 의 지연보다 2 배 이상 큰 지연을 보였다. PT 그룹과 calico 그룹은 처리율에서도 2 배 이상의 차이를 보였고, 특히 PT-flat 은 calico-vxlan 의 처리율 대비 약 4 배의 처리율을 보여주었다. 따라서 어플리케이션 수준에서도 VM 네트워크보다는 CNI 플러그인이 전체 네트워크의 성능에 미치는 영향이 더 크다는 것을 확인할 수 있었다.

일련의 실험 결과를 정리해보면, PT-flat, PT-vxlan, calico-flat 및 calico-vxlan 순으로 네트워크 성능이 높다. PT-flat 이 성능면에서 가장 우수하긴 하나 VM 네트워크로 Flat 네트워크를 사용할 경우 물리 네트워크와의 의존성이 발생하여 유연한 네트워크 운용이 힘들다. 또한 컨테이너 네트워크를 pass-through 하는 경우 성능은 눈에 띄게 좋아졌으나, Kubernetes 에서 제공하는 다양한 네트워킹 모델 (POD 간 통신, Service IP 통신, NodePort 통신)을 사용할 수 없어 실제 상용 환경에 적용하기 힘들다. 따라서 경쟁력 있는 저지연 엣지 컴퓨팅 서비스를 위해서는 최적의 성능을 위해 PT-flat 방식을 채택하되, 상술한 문제점을 해결할 수 있도록 개량된 CNI 플러그인의 추가적인 개발이 필요하다.

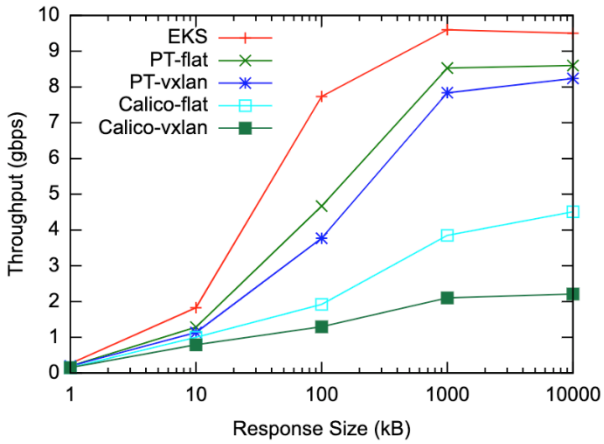


그림 6 Apache JMeter 를 이용한 처리율 측정 결과

Ⅳ. 결론 및 향후 연구

본 연구에서는 엣지 컴퓨팅에서 사용하기 위해 구축한 CoV 형상을 토대로 VM 네트워크와 컨테이너 네트워크의 스택을 조합하여 만든 상이한 네트워크 모델에서 네트워크 처리율을 측정, 비교하고 VxLAN 과 calico CNI 플러그인에서 패킷을 처리하면서 지연되는 시간을 추정했다.

CoV 형상에서 네트워크 성능 저하를 발생시키는 근본적인 원인은 VM 과 컨테이너가 독립적인 네트워크 스택을 사용한다는 것이다. 불필요한 듀얼 스택을 사용하게 됨으로써 각 스택에서의 처리 지연이 증가하고, 어플리케이션 수준에서도 처리율의 감소와 지연 시간의 증가로 이어진다. 따라서 향후 연구로 VM 및 컨테이너 네트워크 스택을 하나로 통합한 네트워크 스택을 개발하여 CoV 네트워크 스택의 성능을 향상시킬 계획이다.

참 고 문 헌

[1] Randazzo, Alessandro, and Ilenia Tinnirello. "Kata Containers: An Emerging Architecture for Enabling MEC Services in Fast and Secure Way." 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS). IEEE, 2019.

[2] Shen, Zhiming, et al. "X-Containers: Breaking down barriers to improve performance and isolation of cloud-native containers." Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. 2019.

[3] "GitHub - containernetworking/cni: Container Network Interface - networking for Linux containers," github.com, https://github.com/containernetworking/cni.

[4] Ducastel, A. (2019). Benchmark results of Kubernetes network plugins (CNI) over 10Gbit/s network. https://itnext.io/ benchmark-results-of-kubernetes-network-plugins-cni-over10gbit-s-network-36475925a560

[5] Y. Park, H. Yang and Y. Kim, "Performance Analysis of CNI (Container Networking Interface) based Container Network," 2018 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, 2018, pp. 248-250.

[6] KAPOČIUS, Narūnas. Overview of Kubernetes CNI plugins performance. Mokslas-Lietuvos ateitis/Science-Future of Lithuania, 2020, 12.

[7] RFC 7348: virtual extensible local area network (VXLAN): a framework for overlaying virtualized layer 2 networks over layer 3 networks

[8] "Overlay networking," docs.projectcalico.org, https://docs.projectcalico.org/networking/vxlan-ipip.

[9] "iPerf - The TCP, UDP and SCTP network bandwidth measurement tool," iperf.fr, https://iperf.fr/.

[10] "GitHub - Mellanox/sockperf: Network Benchmarking Utility,", https://github.com/Mellanox/sockperf.

[11] "Apache JMeter - Apache JMeter™," jmeter.apache.org, https://jmeter.apache.org/.

[12] Rodriguez, et al. REST APIs: A Large-Scale Analysis of Compliance with Principles and Best Practices. 21-39. 10.1007/978-3-319-38791-8_2

모바일 앱 성능 테스트 자동화를 위한 클릭 요소 이미지 분석

김수현*, 문현수, 이영석

충남대학교 컴퓨터공학과

shkim950921@o.cnu.ac.kr, munhyunsu@cnu.ac.kr, lee@cnu.ac.kr

Analysis of clickable element images for automated mobile app testing

SooHyun Kim, Hyunsu Mun, YoungSeok Lee*

Chungnam National University

요 약

기존의 성능 측정 및 디버깅을 위한 모바일 어플리케이션 테스팅은 Android나 iOS와 같은 운영체제에 따른 다른 테스트 방법이 필요하거나, 디버깅용 소스코드를 만들어야한다. 소스코드와 운영체제에 관계없는 모바일 어플리케이션 테스팅을 위해서 사용자를 흉내내는 로봇을 만들고자하였다. 로봇은 사용자 입력을 생성할 하드웨어와 입력 위치 및 시기를 결정할 소프트웨어로 구성된다. 로봇을 이용한 테스트는 사용자 입력 생성과 동작 분석기술이 필요하다. 본 논문에서는 모바일 앱 성능 테스트 자동화를 위하여 사용자 입력 생성 시 클릭 위치를 찾기 위해 CNN을 이용한 이미지 분석 방법을 제안한다. 그 결과 정확도는 78.18%, 정밀도 88.08%, 재현율이 63.7%의 결과가 나왔다. 이미지 분석을 이용한 학습 모델은 로봇기반 자동화 테스트를 하는데 사용하였다.

Ⅰ. 서 론

모바일 어플리케이션 테스팅 도구는 어플리케이션의 성능 측정 및 오류검사를 하는 도구이다. Appium[1]은 iOS, Android, Web의 GUI 단위 테스트를 제공하여 컴포넌트마다 테스트를 수행할 수 있다. AppTest.ai[2] 은 웹 기반 모바일 앱 테스트 소프트웨어로, 테스팅하려는 어플리케이션의 .apk 혹은 .ipk 파일을 제공하면 자동으로 버그를 찾고, 동적 액티비티 그래프를 그려 어플리케이션 버튼들의 동작방식을 확인할 수 있다. Firebase Test Lab[3]은 클라우드 기반 앱 테스트 인프라이다. Firebase Test Lab은 어플리케이션의 속도 및 성능 뿐 아니라, 앱의 오류를 통합적으로 검사한다.

기존 모바일 어플리케이션 테스팅 소프트웨어들은 테스트를 위해서 .apk나 .ipk와 같은 바이트코드들이 필요하거나, 테스트를 위한 개별 소스 코드를 만들어야하기 때문에 테스터 사용자들이 제한적이다. 또한, Android나 iOS와 같은 모바일 운영체제에 따라 테스트 방법이 다르기 때문에 운영체제에 따라 다른 버전의 테스트코드를 제작해야 한다. 우리는 소스코드가 필요 없고 운영체제에 관계없는 어플리케이션 앱 테스트를 위하여 로봇을 이용한 앱 테스트 도구를 만들고자 하였다.

로봇을 이용한 앱 테스트 도구를 만들기 위해서, 사용자 입력 생성과 어플리케이션 동작 분석 기술이 필요하다. 본 논문에서는 사용자 입력 생성을 위해서 클릭 가능한 이미지를 분석 하였다. 이미지 분석을 이용한 터치 요소 분석결과, 정확도 78.18%, 정밀도 88.08%, 재현율 63.74%의 결과를 보였다. 이는 자동 터치를 위한 모바일 어플리케이션 테스트 방법에 기여할 수 있다.

1) http://appium.io/
2) https://apptest.ai/
3) https://firebase.google.com/docs/test-lab

Ⅱ.관련 연구 분석

모바일 어플리케이션 자동화 테스트 방법과 자동화 테스트는 지속적으로 연구되고 되고 있으며[1, 2], 액티비티 구분을 위한 이미지 분석 연구도 진행되고 있다. Ariel Rosenfeld는 안드로이드 어플리케이션의 기능 테스트를 자동화하기 위하여 머신러닝을 이용한 액티비티 분류를 제안하였다[3]. 하나의 어플리케이션은 유사한 구조의 액티비티 구조를 공유한다는 가정을 기반으로 일곱 가지의 액티비티로 구분하였다. 각 액티비티 별로 테스트 케이스를 미리 정의해두고, 테스트 케이스를 기반으로 기능검사를 진행하였다. 액티비티를 자동으로 구분하기 위하여 여섯 가지의 머신러닝 알고리즘을 이용하였으며, KStar 알고리즘을 사용하여 최대 86%의 정확도로 액티비티를 구분하였다.

Android와 iOS등 모바일 운영체제에 관계없는 어플리케이션을 테스트하기 위해서 로봇을 이용한 테스트도 이루어지고 있다[4, 5, 6]. 로봇을 이용한 어플리케이션 테스트를 하려면 동영상 및 이미지 분석을 통하여 어플리케이션 내 요소들을 분석해야한다. Mihai Craciunescu는 로봇을 이용하여 어플리케이션을 테스트하기 위하여 이미지 인식 알고리즘을 이용한 휴대폰의 화면분석 방법을 제안하였다[5]. 화면 분석 방법은 특정 어플리케이션의 아이콘 혹은 어플리케이션 내 특정 아이콘이 주어지면 모바일 화면 내에 해당 아이콘을 찾는 방식으로 분석하였다. 휴대폰 화면 분석을 위해 크

기와 회전에 무관한 이미지 분석 알고리즘인 SURF와 SIFT를 사용하였다. 세 가지 모바일 휴대폰에서 정밀도를 분석한 결과, 최대

Ⅲ. 모바일 앱 성능 테스트 자동화를 위한 클릭 요소 이미지 분석 방법

그림 1 전체 프로그램 구성도

SURF 77.59%, SIFT 24.31%로 모바일 화면 내 아이콘을 찾아낼 수 있었다.

Jih-Gau Juang은 로봇을 이용한 모바일 테스트에 세 개의 카메라를 설치해 카메라로 글씨를 구분하는 연구를 진행하였다[6]. 하나의 카메라에는 색조, 채도, 밝기를 변형하며 글씨 패턴 매칭을 하였고, 두 개의 카메라는 좌, 우를 구분해주는 역할을 하며 3차원적인 화면에 나타나도록 하였다. 1부터 9까지의 문자를 구분해주는 실험을 진행한 결과, 3의 문자를 구분하는 것을 제외하고는 모두 100%의 성공률을 보였으며, A부터 Z까지의 문자를 구분한 테스팅 결과 약 95%의 정확도를 보였다.

어플리케이션 성능 및 기능 테스팅 외에도 개인정보가 노출 될 수 있는 민감한 아이콘을 찾는 연구에서도 이미지 분석을 이용하였다. Xusheng Xiao는 UI에서 사용자 퍼미션이 필요한 민감한 개인정보 데이터가 유출될 수 있는 모든 위젯들을 구분하는 연구를 진행하였다[7]. 민감한 개인 정보들을 8개의 카테고리로 구분하였다. 이미지와 텍스트 분석을 이용하여 각 카테고리에 해당하는 이미지 버튼들을 이미지 분석 알고리즘을 이용하여 유사한 이미지들을 분류하였다. UI layout 파일 분석을 이용하여 유사한 이미지들 및 텍스트의 위치를 찾아내었다. 이미지 분석만 했을 82.4%의 정밀도로 개인정보에 민감한 아이콘을 찾아 낼 수 있었고, 텍스트 분석을 병행 했을 경우 2.01배 향상하였다.

기존 논문에서는 앱 테스트를 위한 이미지 분석 및 머신러닝 기술을 사용하였다. 자동 테스트를 위해 클릭 요소를 찾는 것이 아니라, 어플리케이션에서 특정 이미지를 찾거나 글씨를 인식하는 부분적인 이미지 인식을 통해 테스팅이 가능하다는 것을 보였다. 본 논문에서는 어플리케이션 내 이미지분석을 통하여 클릭 요소와 클릭 불가 요소들을 모두 인식하여 클릭 요소를 찾는 방법을 제안하고자 하였다.

본 연구에서는 Android와 iOS에서 동일한 UI가 적용된 어플리케이션 실험을 가정하였다. 그림 1은 모바일 앱 성능 테스트 자동화를 위한 클릭요소 이미지분석의 전체 과정을 나타내었다. 클릭 요소를 분류하여 실험을 위해 세 가지 단계를 거쳐서 학습하였다. 클릭 요소 이미지를 수집하기 위해 안드로이드 adb를 사용하여 어플리케이션 스크린 샷과, 스크린 샷 에 해당하는 레이아웃 xml을 수집하였다. 수집한 xml 분석하여 클릭 요소와 클릭 불가 요소들의 좌표를 파악하고, 스크린 샷의 좌표에 해당하는 이미지로 잘라 각 요소별로 이미지들을 저장하였다. 저장한 이미지 데이터들은 CNN을 통해 클릭 요소와 클릭 불가 요소의 특징을 뽑아 학습하였다.

1. 데이터 수집 방법

CNN을 이용한 클릭요소와 클릭 불가 요소를 구분하기 위해서 각 클릭 가능 여부에 따른 이미지 데이터 셋 수집이 필요하다. 안드로이드의 adb를 이용하여 어플리케이션의 레이아웃 정보들을 수집하였다. 어플리케이션의 레이아웃 정보는 xml dump를 사용하여 어플리케이션 액티비티에 해당하는 xml과, 어플리케이션 액티비티에 해당하는 스크린 샷을 수집하였다.

표 1 수집한 액티비티 개수

	개수 (개)
어플리케이션 1개당 xml수	10
어플리케이션 1개당 screen shot 수	10
데이터 수집을 위한 어플리케이션 수	76
총 액티비티 개수	760

다수의 xml과 스크린 샷을 수집하기 위해서 adb의 UI-Automator 명령어를 이용하여 자동으로 클릭 요소들을 찾아 클릭 후. 변경된 액티비티의 xml과 스크린 샷을 저장하였다. 어플리케이션 수집 결과, 표 1에서 제시된 어플리케이션처럼 총 76개 어플리케이션에 각각 10개의 터치 이벤트를 실행하여 10개의 액티비티 정보를 수집하였다. 수집한 76개의 어플리케이션은 표 2 처럼 구글

표 2 카테고리 별 어플리케이션 개수

카테고리	개수	카테고리	개수
Game	6	Photography	2
Beauty	1	Productivity	13
Business	1	Shopping	4
Comics	1	Social	2
Communication	5	Tools	4
Education	3	Travel&Local	6
Entertainment	2	Video Player&Editor	1
Finance	2	Lifestyle	9
Food&Drink	2	Map& Navigation	2
Health&Finess	5	Music&Audio	2
Library&Demo	1	News&Magazine	2
		계	76

플레이스토어 기준으로 22개의 카테고리로 분류하였다. 어플리케이션 중 사람들이 많이 사용하는 어플리케이션[4]인 넷플릭스는 Entertainment에, 유튜브는 Video Player & Editor에 포함되고, 인스타그램, 페이스북은 Social에 카카오 톡은 채팅 어플리케이션은 Communication에 포함된다.

수집한 어플리케이션의 클릭 가능한 요소와 클릭 불가능한 요소는 카테고리에 따라 다른 특징을 보인다. News & Magazine은 텍스트가 다른 어플리케이션들에 비해 상대적으로 많은 비율을 차지하지만 클릭요소는 상대적으로 적게 나타난다. Beauty와 Shopping은 다른 어플리케이션에 비해 많은 이미지 버튼을 포함하며, 채팅 어플리케이션은 이미지와 글자가 함께 포함된 리스트 클릭이 다른 어플리케이션에 비해 상대적으로 많을 수 있다. 따라서, 어플리케이션 별로 카테고리를 분류하였다.

리스트 1 안드로이드 xml 노드 속성 리스트

클릭 가능 요소	
• checkable	노드가 체크 가능한가?
• checked	노드가 체크되어있는가?
• clickable	노드를 클릭했을 때 클릭 이벤트가 적용되어 있는가?
• selected	노드가 선택되어있는가? 선택된 노드가 하이라이트 되어있는가?
• long -clickable	해당 노드를 길게 눌렀을때에 대한 클릭 이벤트가 있는가?
클릭 불가 요소	
• scrollable	상하좌우로 화면이동이 가능한가?
• enabled	현재 노드 및 하위 클래스노드의 상태를 변경할 수 있는가?
• password	노드의 입력내용이 숨겨지는가?
• focusable	해당 노드가 Focus를 가질 수 있는가?
• focused	해당 노드가 Focus되어 있는가?

수집한 어플리케이션의 xml은 계층구조로 되어있고, 각 계층은 노드로 구성되어 있다. 각 정보들은 클릭, 체크등 사용자와 상호작용할 수 있는 속성들을 가진다. 노드의 최 하단에는 bounds로 해당 노드가 어플리케이션에 어디에 위치하는지 알 수 있다. 그림 1 상단에 나와 있는 노드는 Frame Layout에 대한 노드로, 왼쪽 스크린 샷 전체 부분에 해당된다. 이 bounds값들은 스크린 샷 이미지를 클릭요소 및 클릭 불가요소 별로 자르기 위하여 사용하였다.

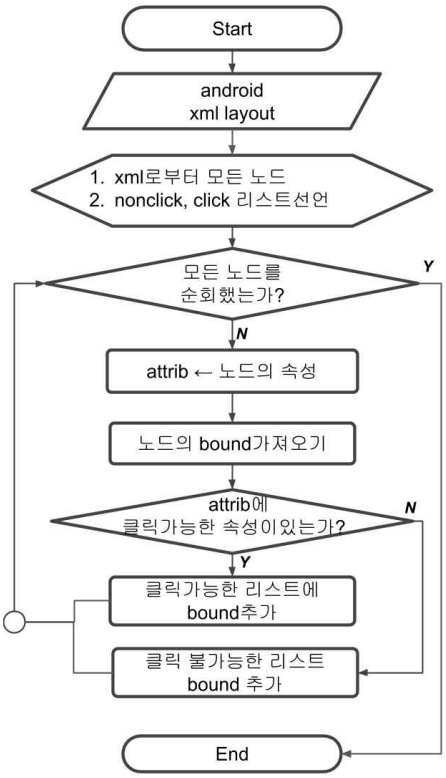
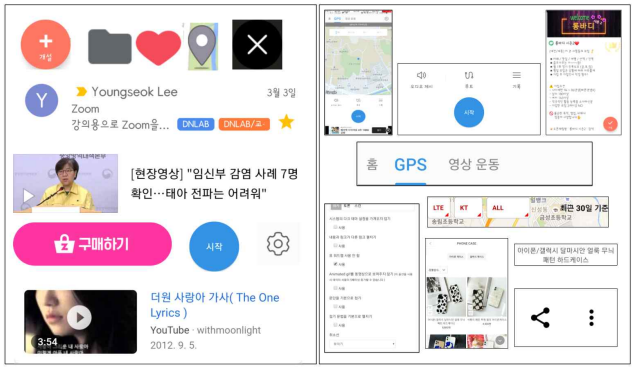


그림 2 클릭요소의 좌표를 찾는 흐름도

그림 2는 xml의 노드에서 클릭 요소를 결정짓는 속성들을 모두 가지고 올 수 있는 순서도이다. 클릭 요소들을 모두 찾아내기 위해 10개의 클릭요소의 리스트 1에 제시된 속성에 근거하여 checkable, checked, clickable, long-clickable, selected 다섯 개의 속성을 클릭 가능한 데이터로 두었다. enable는 리스트 1에 제시되어 있는 모든 속성들을 변경시킬 수 있는지 설정해주는 것으로, 노드 하나에 대한 상태변경을 말하는 것이 아니므로 제외하였다. scrollable역시 액티비티 전체이미지를 나타내는 Layout에 적용되는 것으로, scrollable 속성이 들어간 노드들은 클릭 가능한 요소로 식별하기 어렵다. 마찬가지로, focusable과 focused, password들은 특정 뷰 에만 적용되므로 액티비티의 클릭 가능한 요소들로부터 제외하였다.

수집한 클릭 요소가 저장된 좌표는 csv로 저장하였다. 파일에 어플리케이션의 스크린 샷, 노드에서 가져온 bounds값 인 x. y와 clicklist를 저장하였다. 저장한 csv를 읽어 어플리케이션에 해당하는 스크린 샷에서 이미지를 잘라 만약 clicklist에 속성이 존재한다면 클릭 가능한 이미지로, 존재하지 않는다면 클릭 불가능한 이미지로 저장하였다. 그림 2는 클릭 가능한 액티비티와 클릭가능하지 않는 액티비티를 잘라서 부분 저장해놓은 결과이다. 클릭 가능한 이미지들은 대부분 아이콘이나, 리스트 뷰 형태의 이미지들이고, 클릭 가능하지 않은 아이콘들은 아이콘들 중 비 활성화되어 있거나, 전체화면, 텍스트 뷰 등이 있었다.

4) https://play.google.com/store/apps/top



<Clickable Element> <Non-Clickable Element>

그림 3 클릭 가능한 요소와 클릭 불가능한 요소의 샘플

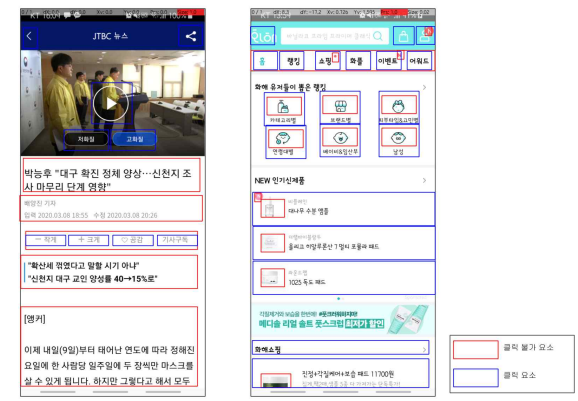
2. 학습 및 결과

표 3 전체 어플리케이션에 대한 정확도

지표	(%)
정확도	78.18
정밀도	88.08
재현율	63.74
F1 score	73.96

잘라진 이미지는 클릭 요소와 클릭 불가 요소 두 가지 라벨로 분류하였다. 터치 가능한 이미지 총 9,832개, 터치 불가능한 이미지 35,202개로 하여 총 45,034개를 학습데이터로 이용하였다. 학습 데이터는 그림 2와 같이 다양한 크기의 이미지를 가로, 세로 39 x 39로 이미지 사이즈를 줄여 분석하였다.

수집한 데이터는 와 같은 레이어를 구성하여 모델을 만들고, CNN을 이용하여 학습하였다. CNN 모델은 2개의 3x3 컨볼루션 필터를 적용하여 서로 다른 클릭 가능한 요소들의 특징을 추출 하였다. 추출한 특징들을 flattern을 이용하여 단일 차원으로 축소하였고, 두 개의 dense 레이어를 이용하여 레이블을 2개로 줄이면서 클릭 요소를 구분하였다.



<클릭요소와 클릭 불가 요소가 뚜렷한 예> <클릭요소와 클릭 불가 요소가 뚜렷하지 않은 예>

그림 4 클릭요소가 뚜렷하게 나타난 앱 화면과 뚜렷하게 나타나지 않은 예

어플리케이션으로부터 클릭 가능한 요소들을 수집하여 20번 반복하여 모델을 제작하였다. 표 3에서 학습 정확도 78.18%, 손실률은 56%가 나왔다. 정밀도는 88.08%로 클릭 요소인 것들은 정확히 찾아낼 수 있었다. 그러나 재현율은 63.74%로 상대적으로 낮아 클릭요소가 아닌 것도 클릭 요소로 찾아지는 경우도 있었다.

그림 4는 클릭요소가 뚜렷하게 나타난 어플리케이션과 클릭 요소가 뚜렷하게 나타나지 않는 어플리케이션 사례를 보인 그림이다. 파란색 상자는 클릭요소를, 붉은색 상자는 클릭 불가 요소를 나타낸다. 그림 4의 좌측에 있는 뉴스와 같이 텍스트가 많이 나오는 어플리케이션 종류는 클릭요소와 클릭 불가능 요소가 뚜렷하게 나타났다. 반대로, 클릭 요소와 클릭 불가 요소가 뚜렷하게 나타나지 않는 어플리케이션은 중첩된 레이아웃들이 많은 어플리케이션이였다. 이 어플리케이션은 클릭 불가능 요소에 대해서는 뚜렷한 특징이 보이지 않아 재현율은 정밀도에 비해 상대적으로 낮게 나왔다. 이는 그림 4의 오른쪽 그림처럼 클릭요소 안에 클릭 불가 요소들이 나오는 경우, 클릭 요소와 클릭 불가 요소가 뚜렷한 차이가 나지 않고, 클릭 요소로 인식 하는 경우가 많기 때문으로 보인다.

Ⅲ. 결론 및 향 후 연구

본 논문에서는 이미지 분석을 이용한 어플리케이션의 터치요소 식별방법을 위한 자동 데이터 수집 방법 및 수집 데이터를 활용한 학습 방법과 그 활용에 대해서 보였다. 데이터 수집은 안드로이드의 adb를 이용하여 xml과 스크린 샷 분석을 통해 클릭 요소와 클릭 불가요소를 구분했다. 수집한 데이터는 CNN을 이용하여 학습을 하였다. 어플리케이션 터치 요소의 정확도는 78.18%, 재현율 63.74%, 정밀도 88.08%가 나왔다. 이는 어플리케이션 테스트 자동화에 활용할 수 있다.

참 고 문 헌

[1] 이정규, 국승학, 김현수. (2008). 시나리오의 자동 생성을 통한 GUI 테스트 케이스 생성. 한국정보과학회 학술발표논문집, 35(1A), 62-63.

[2] 김하연, 노혜민, 유철중. (2019). 모바일 애플리케이션 GUI 테스팅을 위한 GUI 이벤트 추출 크롤링 기법. 한국정보기술학회 논문지, 17(12), 135-145.

[3] Rosenfeld, Ariel, Odaya Kardashov, and Orel Zang. "Automation of android applications functional testing using machine learning activities classification." Proceedings of the 5th International Conference on Mobile Software Engineering and Systems. 2018.

[4] Mao, Ke, Mark Harman, and Yue Jia. "Robotic testing of mobile apps for truly black-box automation." IEEE Software 34.2 (2017): 11-16.

[5] M. Craciunescu, S. Mocanu, C. Dobre and R. Dobrescu, "Robot Based Automated Testing Procedure Dedicated to Mobile Devices," 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP), Maribor, 2018, pp. 1-4.

[6]Juang, Jih-Gau, and I-Hua Cheng. "Application of character recognition to robot control on smartphone test system." Advances in Mechanical Engineering 9.3 (2017): 1687814017693181.

[7] Xiao, Xusheng, et al. "Iconintent: automatic identification of sensitive ui widgets based on icon classification for android apps." 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). IEEE, 2019.

A Resource Optimization Framework for Federated Learning Over Wireless Networks

Latif U. Khan, Umer Majeed, Choong Seon Hong
Department of Computer Science and Engineering, Kyung Hee University, Yongin, 446-701 Korea
Email: {latif, umermajeed, cshong}@khu.ac.kr

Abstract

In the past few decades, a significant increase in number of Internet of Things (IoT) devices has been observed which generate huge amount of data. This data can be used for the training of different machine learning models to make different applications smart. However, the traditional machine learning model has a disadvantage of moving the user data to a centralized server which causes serious privacy issues. Coping with this issue, federated learning (FL) is a promising scheme to enable machine learning without transferring the data to a centralized server. In this paper, we propose software-defined networking (SDN) based architecture for FL over wireless networks. We propose a framework that consider joint user association and resource allocation in a heterogeneous cellular network for improving the global FL model accuracy. Furthermore, we consider a hybrid control plane for SDN to offer scalability which will be one of the key requirement of future networks.

1. Introduction

A rapid rise in the Internet of Things (IoT) enabled smart environments has been witnessed recently [1][2]. These devices produce a significant amount of data that must be effectively used to train machine learning models for making the applications smart. Different machine learning schemes can be used to make these applications smart; however, traditional machine learning has serious privacy concerns. To tackle this challenge of privacy concern in traditional machine learning, we can use federated learning (FL). FL offers distributed learning which involves the training of local model at the devices and then sending the local model parameters of all the devices to a centralized server. The centralized server after global model aggregation sends back the global model parameters to the devices. This transfer of learning model parameters between the local devices and centralized serves takes place in an iterative manner until a certain level of accuracy has been attained [3][4].

Although FL offers significant advantages it uses significant communication resources during the training process. Therefore, it is necessary to optimize the communication resources for FL over cellular networks. On the other hand, the packet error rate has a proportional effect on the performance of FL [5]. Additionally, the significant increase in the number of IoT devices is expected in the foreseeable future. Software-defined networking (SDN) and network function virtualization (NFV) are considered to the basic building blocks for 5G and beyond networks. The operation of SDN is based on a separation of the control plane from the data plane and thus, enables efficient and easier management of the network [5]. NFV allows the use of generic hardware for implementing different network functions on virtual machines in a cost-effective way [6].

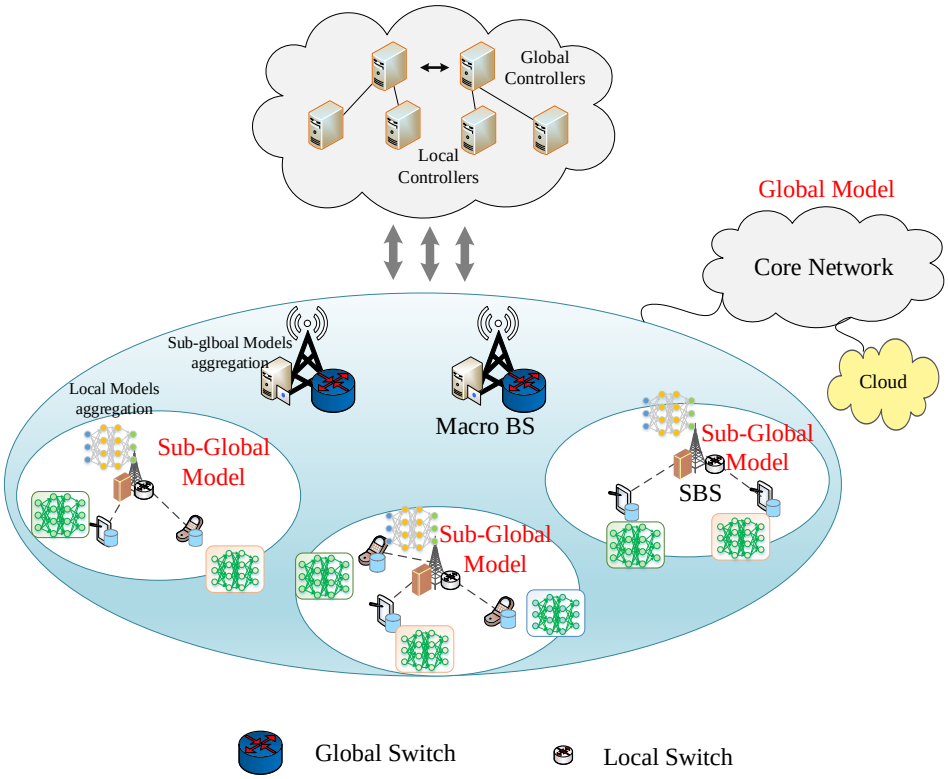


Fig 1: Proposed framework for federated learning

Motivated by the above-discussed factors, we propose an architecture based on SDN and NFV to enable effective FL for wireless networks in a resource optimized way.

2. Proposed Framework

Consider a heterogeneous cellular network consisting of macro-cell base stations (MBSs) and small cell base stations (SBSs). In [6], a hierarchical fashion of FL is adopted to enable resource-efficient operation. Similarly, we adopt the hierarchical fashion for learning for a global FL model. First of all, sub-global models are computed at every SBS in an iterative fashion which is followed by global model computation at the core network. On the other hand, the packet error rate significantly determines the performance of FL [5]. The SBS preferably will not consider the local learning models of the devices having a higher packet error rate. Therefore, it is necessary to reduce the packet error rate for improving the performance of FL over wireless networks [5]. Increasing the throughput of the devices involved in learning reduces the packet error rate and thus, can improve the FL model accuracy. One of the promising ways to improve the network throughput is through joint optimization of devices association with base SBS and resource allocation. For association and resource allocation, we can use iterative algorithms [7][8]. The iterative algorithms can further use matching theory and other heuristic algorithms specifically for association and resource allocation [9]. To improve the FL model accuracy over the wireless network, we propose a framework based on SDN as shown in Fig. 1. The proposed framework consists of a data plane and a control plane. The data plane consists of radio access networks, mobile devices, edge computing servers, cloud computing servers, and SDN switches. In our proposed framework, we use two types of SDN switches: Local switches and global switches. The local switches are placed near the devices at the SBS, whereas the global switches are positioned at the MBS and core network. Every global switch can be connected to multiple local switches. The local switches can communicate with each other through global switches, whereas global switches can communicate with each other directly.

In a control plane, there are two types of controllers: Global controllers and local controllers. The reason for

using two types of controllers is to enable scalability which is one of the primary design objectives of foreseeable future wireless networks. The job of the global controller is local switches management, radio resource management, cloud management, quality of service (QoS) management, among others. The most important role of global controllers in our proposed framework is to control joint devices association and resource allocation. All these functions are based on the requirement of global network-wide view information.

On the other hand, local controllers are used to control local switches that require only local information for their operation and do not need global network-wide view information. The local controllers manage the process of the local devices learning which is followed by sending of model parameters to SBS equipped with edge computing servers. The SBSs are connected using high-speed backhaul links to MBSs.

3. Discussion

We have proposed an architecture based on SDN to enable hierarchical FL for heterogeneous cellular networks in a resource optimized fashion. One of our main contributions is the use of a hybrid control plane to enable scalable operation. Using our proposed algorithm, we can achieve resource optimized and scalable FL over wireless networks to enable different smart applications.

4. Conclusions

In this paper, we have proposed an architecture that enables resource-efficient and scalable FL over wireless networks. It is concluded that SDN with a hybrid control plane can be used effectively in FL processes to offer effective FL in a resource optimized way. Furthermore, the hybrid control plane has the advantage of a more scalable operation for coping with the increasing number of users in the foreseeable future.

As future work, we can formulate an optimization problem for joint device association and resource optimization for hierarchical FL in heterogeneous cellular networks which will be followed by an appropriate solution.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing) *Dr. CS Hong is the corresponding author.

REFERENCES

[1] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. Kazmi, T. N. Dang, and C. S. Hong, “Edge Computing Enabled Smart Cities: A Comprehensive Survey,” *arXiv preprint arXiv:1909.08747*, 2019.

[2] W. Saad, M. Bennis, M. Chen, “A vision of 6G wireless systems: Applications, trends, technologies, and open research problems,” *arXiv preprint arXiv:1902.10265*, 2019.

[3] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, C. S. Hong, “Federated Learning over Wireless Networks: Optimization Model Design and Analysis,” *IEEE Conference on Computer Communications*, pp. 1387-1395, 2019.

[4] L. U. Khan, N. H. Tran, S. R. Pandey, W. Saad, Z. Han, M. N. H.Nguyen, and C. S. Hong, “Federated learning for edge networks: Resource optimization and incentive mechanism,” arXiv preprint arXiv:2917118, 2019.

[5] M. Chen, Z. Yang, W. Saad, C. Yin, H.V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *arXiv preprint arXiv:1909.07972* (2019).

[6] M.S.H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," *arXiv preprint arXiv:1909.02362* (2019).

[7] Y. Gu, W. Saad, M. Bennis, M. Debbah, and Z. Han, "Matching theory for future wireless networks: Fundamentals and applications," *IEEE Communications Magazine,* vol. 53, no. 5, pp. 52-59, 2015.

[8] L. Song, D. Niyato, Z. Han, and E. Hossain, "Game-theoretic resource allocation methods for device-to-device communication," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 136-144, 2014.

[9] S. M. A. Kazmi, N. H. Tran, W. Saad, Z. Han, T. M. Ho, T. Z. Oo and C. S. Hong, "Mode selection and resource allocation in device-to-device communications: A matching game approach," *IEEE Transactions on Mobile Computing,* vol. 16, no. 11, pp. 3126-3141, 2017.

네트워크 트래픽 발생량 분석을 활용한 연결 토폴로지 탐색

박지태, 구영훈, 백의준, 신무곤, 김명섭
고려대학교

{pjj5846, gyh0808, pb1069, tm0309, tmskim}@korea.ac.kr

A Connection Topology Discovery using Generated Network Traffic Analysis

Jee-Tae Park, Young-Hoon Goo, Ui-Jun Baek, Mu-Gon Shin, Myung Sup Kim
Korea Univ.

요 약

최근 과학 기술의 발전으로 인한 네트워크 환경을 발전으로 다양한 응용과 대용량의 트래픽이 발생하고 있다. 이러한 추세에 따라 네트워크 트래픽 과부하, 외부 악성 행위와 같은 네트워크 장애 현상이 빈번하게 발생하고 있다. 네트워크 장애 현상을 신속하게 탐지하고, 원인 분석 및 예방하기 위해서 네트워크 장비들의 연결 토폴로지에 대한 직관적인 정보가 필요하다. 하지만 기존의 네트워크 토폴로지 탐색하는 방법은 관리자가 수동적으로 관리 하기 때문에 많은 시간과 노력이 들며, 제한된 장비 만을 탐색 하거나, 연결을 탐색하더라도 연결 포트 정보는 탐색하지 못한다는 문제점이 있다 따라서 본 논문에서는 연결이 정의 된 네트워크 장비를 대상으로 장비의 네트워크 트래픽 발생량 분석을 통해 연결 포트를 탐색하는 방법을 제안한다. 제안하는 방법을 검증하기 위해서 실제 네트워크 장비에 대해 실험을 진행하였으며, 실험 결과의 검증을 통해 본 논문의 타당성을 증명한다.

Ⅰ. 서 론 [1]

오늘 날의 인터넷은 네트워크 고속화 및 과학 기술의 급격한 발전으로 다양한 응용과 대용량의 트래픽이 발생하고 있다. 이에 따라 네트워크 트래픽 과부하, 네트워크 악성 행위와 같은 다양한 네트워크 장애 현상이 발생하고 있다. 네트워크 장애 현상에 신속하게 대응하기 위해서 네트워크 관리 시스템(Network Management System)을 구축하여 여러 가지 네트워크 결함에 대한 분석이 필요하다[1,2]. 현재 네트워크 관리 시스템은 네트워크 자원 관리, 서버 관리, 장애 현상 원인 분석 및 예방의 기능 제공하며, 이러한 기능들을 효율적으로 관리하기 위해서 관리자는 네트워크 연결 토폴로지에 대해 알아야한다[1].

관리자가 네트워크 장비들의 연결 구성을 수동으로 관리 할 경우, 많은 시간과 노력, 비용이 든다. 따라서 네트워크 토폴로지 탐색을 위해 여러 가지 방법론 연구되어 왔으며, 가장 널리 알려진 방법이 Traceroute 이다[2].

Traceroute 를 활용한 방법은 쉽고 편리하다는 장점이 있지만, 목적지 중복 탐색 및 대규모의 네트워크에서 사용하기 어렵다는 문제점이 있다. Traceroute 이외의 기존 방법들 역시 제한된 계층의 장비만을 탐색하거나, 연결 포트 정보를 탐색 할 수 없거나, 시간 및 리소스의 낭비가 심하다는 문제점이 존재한다.

따라서 본 논문에서는 이러한 문제점을 해결하기 위해서 SNMP 를 활용하여 네트워크 장비 MIB 정보를 저장하고, 저장 된 정보를 활용하여 트래픽 발생량 분석 및 네트워크 토폴로지를 탐색하는 방법을 제안한다.

본 논문의 구성은 다음과 같다. 1 장 서론에서 네트워크 토폴로지 탐색 연구 필요성과 기존 연구의 문제점을 설명하고, 2 장 관련 연구에서는 기존에 존재하는 여러 가지 네트워크 장비 탐색 방법론에 대해 언급하고, 각 방법론에 대한 장단점을 분석하였다. 3 장 본론에서 제안하는 토폴로지 탐색 알고리즘 설명하고, 4 장에서 실제 네트워크 장비에 대한 실험을 진행한다. 마지막으로 5 장에서 결론 및 향후 연구를 언급하고 마친다.

이 논문은 2018 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구 사업이며(NRF-2018R1D1A1B07045742), 2018 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539-002, 블록체인의 트랜잭션 모니터링 및 분석 기술개발)

Ⅱ. 관련 연구

대부분의 네트워크 관리 시스템에서는 네트워크 자동 탐색 기능을 활용하여 네트워크 장비들의 연결 구성과 연결 변화에 대해 신속하게 모니터링하고 있다. 하지만 네트워크 자동 탐색 기능은 대부분 3 계층까지의 장비들만 탐색하기 때문에 2 계층의 네트워크 장비에 대한 관리 방안이 필요하다.

최근에 이러한 문제를 해결하기 위해 여러 방법론이 연구되어 왔으며, 가장 잘 알려진 방법은 Traceroute 를 활용한 방법이다.

먼저 Traceroute 를 활용한 방법은 출발지에서 목적지까지의 경로를 식별하기 위해 패킷을 보낸다. 이 때, 패킷 헤더의 TTL(Time-To-Live) 값을 조정하여 경로 상에 있는 네트워크 장비를 식별하고, 해당 경로에 대한 정보를 수집한다. 하지만 네트워크 장비 수가 많은 기업용 혹은 대규모의 네트워크를 대상으로 적용할 경우, 시간이 오래 걸린다는 문제점이 있다[2]. 또한, 출발지와 도착지가 다르더라도, 이전에 탐색한 중복되는 경로를 탐색하기 때문에, 비효율적이며, 네트워크 부하가 많이 걸린다[1,5].

따라서 최근에는 LLDP 및 BDDP 와 같은 2 계층 네트워크 장비 탐색 방안이 연구되고 있다[3]. LLDP 는 IEEE 802.1AB 에서 표준으로 정의 된 프로토콜로, LLDP 를 활용하여 연결 된 2 계층의 장비 연결 정보를 알 수 있다. 하지만 LLDP 는 또한, 장비 간 연결된 포트 정보는 직접 연결 된 장비 간의 연결 정보만 알 수 있으며, 오래 된 네트워크를 대상으로 적용 할 수 없다는

단점이 있다. BDDP 는 LLDP 와 마찬가지로 2 계층 장비의 연결 정보를 편리하게 알 수 있다. 그리고 오픈 소스로 구현되어 있기 때문에 쉽게 사용 할 수 있다는 장점이 있다. 하지만 BDDP 는 브로드캐스트 형태로 패킷을 전송하기 때문에 시간과 리소스를 많이 소모하며, LLDP 와 마찬가지로 오래된 네트워크를 대상으로 사용하기 어렵다는 단점이 있다[3].

이러한 문제점을 해결하기 위해서 SNMP 기반의 네트워크 장비 탐색 방법이 연구되고 있다[4]. SNMP 를 활용하면 서버에서 대상 에이전트의 여러 정보를 쉽게 알 수 있다. SNMP 기반의 네트워크 장비 탐색 방법은 이러한 정보를 취합하여 네트워크 토폴로지를 형성한다. SNMP 를 활용한 방법은 다른 방법에 비해 쉽고, 편리하며, 여러 계층의 장비들에 대해 적용 가능하기 때문에 범용성이 높다[4]. 하지만 기존의 SNMP 기반 탐색 방법은 BRIDGE MIB 정보에 대한 의존도가 매우 크다. BRIDGE MIB 정보는 BRIDGE TABLE 로 구성되어 있으며, 부정확하게 구성 될 가능성이 있으며, 테이블 갱신 간격이 길어질수록 2 계층 장비를 탐색하기 어렵다는 문제점이 존재한다.

따라서 본 논문에서는 SNMP 기반 네트워크 장비 탐색의 장점을 유지하면서 문제점을 해결하기 위해 새로운 탐색 알고리즘을 제안한다. 제안하는 알고리즘은 BRDIGE MIB 정보를 사용하지 않기 때문에 존재하는 여러 문제점을 해결 할 수 있다.

Ⅲ.. 본론

제안하는 방법은 연결이 정의 된 장비 들을 대상으로 연결 포트를 탐색하는 방법이다. 연결 포트를 탐색하기 위해서 연결 된 장비 간에 발생하는 트래픽 양의 비교한다

이 때, SNMP MIB 정보로 ifInOuctet, ifOutOuctet 을 사용하며, 해당 장비에서 각 포트 별 발생하는 트래픽 양을 알 수 있다. ifOutOuctet 에 대한 예시는 그림 1 에 나타나있다. 그림 1 에서 대부분의 인덱스는 연결 되어있지 않기 때문에 트래픽이 발생하지 않으며, 14, 21, 28 인덱스에서 트래픽이 발생하고 있다. 이 때, 발생한 트래픽 양을 비교하여 같으면 인덱스와 맵핑 된 포트를 찾는다.

```
[                    ]# snmpwalk -v 2c -c                    1.3.6.1.2.1.2.2.1.16
IF-MIB::ifOutOctets.1 = Counter32: 0
IF-MIB::ifOutOctets.2 = Counter32: 0
IF-MIB::ifOutOctets.3 = Counter32: 0
IF-MIB::ifOutOctets.4 = Counter32: 0
IF-MIB::ifOutOctets.5 = Counter32: 0
IF-MIB::ifOutOctets.6 = Counter32: 0
IF-MIB::ifOutOctets.7 = Counter32: 0
IF-MIB::ifOutOctets.8 = Counter32: 0
IF-MIB::ifOutOctets.9 = Counter32: 0
IF-MIB::ifOutOctets.10 = Counter32: 0
IF-MIB::ifOutOctets.11 = Counter32: 0
IF-MIB::ifOutOctets.12 = Counter32: 0
IF-MIB::ifOutOctets.13 = Counter32: 0
IF-MIB::ifOutOctets.14 = Counter32: 70109012
IF-MIB::ifOutOctets.15 = Counter32: 0
IF-MIB::ifOutOctets.16 = Counter32: 0
IF-MIB::ifOutOctets.17 = Counter32: 0
IF-MIB::ifOutOctets.18 = Counter32: 0
IF-MIB::ifOutOctets.19 = Counter32: 0
IF-MIB::ifOutOctets.20 = Counter32: 0
IF-MIB::ifOutOctets.21 = Counter32: 188594904
IF-MIB::ifOutOctets.22 = Counter32: 0
IF-MIB::ifOutOctets.23 = Counter32: 0
IF-MIB::ifOutOctets.24 = Counter32: 0
IF-MIB::ifOutOctets.25 = Counter32: 0
IF-MIB::ifOutOctets.26 = Counter32: 0
IF-MIB::ifOutOctets.27 = Counter32: 0
IF-MIB::ifOutOctets.28 = Counter32: 1103156
```

Figure 1. ifOutOuctet MIB 정보 예시

하지만 MIB 정보를 저장하는 시간이 다를 때는 한 시점에서 발생하는 트래픽 양이 다를 수 있기 때문에 정확한 연결 정보가 나오지 않는다. 따라서 각 장비에서 10 초 간 발생한 트래픽 양을 비교하여 연결 토폴로지를 탐색한다.

제안하는 방법의 알고리즘은 그림 2 에 나타나있다. 먼저 입력으로 연결 된 장비의 IP 리스트가 들어온다. 입력으로 들어온 연결 된 장비의 IP 에 대해 쌍을 A, B 라고 정할 때, A 의 ifInOuctet, ifInOuctet(10 sec)과 B 의 ifOutOuctet, ifOutOuctet(10 sec)을 을 구한다. 이 때, ifInOuctet(10 sec)는 ifInOuctet 가 저장 되고 10 초 후에 저장 된 ifInOuctet 로, 두 값의 차이를 구하면 해당 인덱스에서 10 초 동안 발생한 트래픽 양을 알 수 있다. 연결 된 두 장비에서 연결 인덱스를 찾으면, ifName MIB 정보를 활용하여 연결 된 포트를 찾을 수 있다.

Algorithm 1 : Device Port Connection Discovery
Input : List of Device IP, MIB Information of Devices, Connected Information
Output : Connection Information
Notation – D : List of Device IP / In : Interface Information
Diff : Difference of ifInOuctet (10 sec) and ifInOuctet or ifOutOuctet (10 sec) and ifOutOuctet / In : Interface Information of the Device

1. Get the IP List of the Devices
2. Set the Connected Device Pair $[D_i, D_j]$
3. **for** i=0 **to** All Devices
4. **for** j=i+1 **to** All Devices
5. Load ifInOuctet of D_i and ifInOuctet (10 sec) of D_i
6. Load ifOutOuctet of D_j and ifOutOuctet (10 sec) of D_j
7. Get the difference of ifInOuctet (10 sec) and ifInOuctet of D_i
8. Get the difference of ifOutOuctet (10 sec) and ifOutOuctet of D_j
9. **if** $Diff_i$ == $Diff_j$
10. Get the Ouctet Index of $[D_i, D_j]$
11. Get the Interface Information of $[D_i, D_j]$
12. Store the In_i, In_j and D_i, D_j

Figure 2. 장비 포트 연결 탐색 알고리즘

최종 결과로 연결 된 장비 타입과, IP 정보, 연결 포트를 알 수 있다. 제안하는 방법에서는 연결 정보를 csv 파일로 저장하여, 한 행에 연결 정보를 저장하였다. 도출한 csv 파일은 데이터베이스를 활용하여 네트워크 관리 시스템에서 연결 토폴로지 UI 정보로 제공 될 수 있다.

Ⅳ. 실험

제안하는 방법을 검증하기 위해 실제 네트워크 장비를 대상으로 실험을 진행하였다. 실험에 사용한 네트워크 장비는 L2 스위치 2 대, L3 스위치 2 대이며, 실험 환경은 그림 3 에 나타나있다.

여러 계층 장비 간의 연결을 확인 하기 위해 그림 3 과 같이 구성하였으며, 호스트에서 제안하는 시스템을 실행한다. 이 때, 설치 된 시스템은 Linux CentOS 7 에서 쉘 스크립트와 C/C++로 구현되었다.

제안하는 방법을 통해 그림 3 의 L2-L2, L2-L3, L3-L3 연결에 대해 정확하게 연결 포트를 탐색할 수 있었다. 탐색 된 연결 및 연결 포트 정보는 그림 4 에 나타나있다. 2 장 본론에서 언급한대로, 저장 된 csv 파일은 한 행에서 연결 된 두 장비의 IP, 포트 정보가 저장된다.

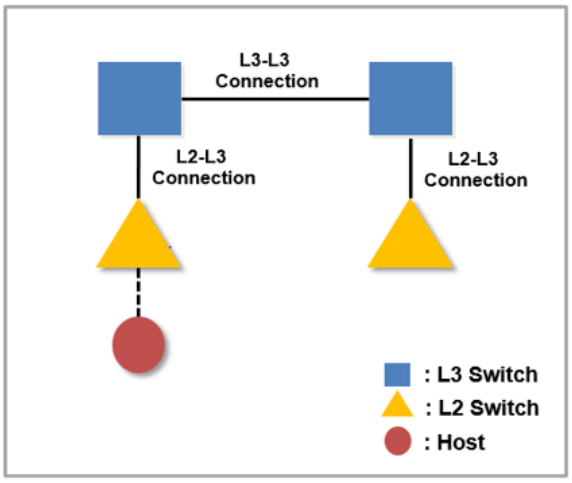


Figure 3. 실험 네트워크 구성도

L3Switch0	Fa0/1	192.168.30.	L3Switch1	Fa0/1	192.168.30.
L2Swtich0	Fa0/2	192.168.20.	L3Switch1	Fa0/1	192.168.20.
L2Switch0	Fa0/1	192.168.20.	L2Switch1	Fa0/7	192.168.20.

Figure 4. 연결 포트 탐색 결과

하나의 토폴로지만으로 제안하는 방법을 검증하기 어렵기 때문에, 네 대의 네트워크 장비로 구성할 수 있는 모든 토폴로지에 대해 실험을 진행하였다. 구성 할 수 있는 토폴로지 수는 4 가지 이며, 각각의 토폴로지에 대해 동일한 실험을 진행하였다. 각 토폴로지에 대한 실험 결과도 마찬가지로 그림 4 와 같이 잘 나오는 것을 확인 할 수 있었다.

V. 결론 및 향후 연구

본 논문에서는 네트워크 토폴로지 탐색 분야에서 기존에 존재하는 여러 방법들을 분석하고 각 방법들의 문제점에 대해서 언급하였다. 그리고 문제점을 해결하기 위해서 SNMP 기반으로 네트워크 장비의 발생 트래픽 양을 비교하고, 이를 활용한 네트워크 토폴로지 탐색 방법을 제안하였다.

3 장 본론에서 제안하는 방법의 상세 알고리즘 제시하고, 4 장 실험에서는 제안하는 알고리즘을 실제 네트워크 장비에 적용하였다. 본 논문에서 제안하는 방법은 기존 방법의 여러 가지 문제점을 해결 가능 할 뿐만 아니라, 알고리즘이 비교적 쉽고 간편하기 때문에 수행 시간도 상당히 줄일 수 있었다.

하지만 본 논문에서 적용한 네트워크는 네 대의 스위치로, 소규모의 네트워크이기 때문에 타당성을 검증하기에 충분하지 않다. 그리고 측정 시간의 차이로 인하여 계산된 수치가 차이가 날 경우, 부정확한 토폴로지가 도출 될 수 있다.

따라서 향후 연구로 여러 네트워크에 적용하여 알고리즘을 보완할 예정이며, 관련 연구에서 언급한 LLDP, BDDP 등의 여러 방법들을 함께 연구하여, 제안하는 방법의 한계점을 해결 할 예정이다.

그리고 결과로 나온 csv 파일의 네트워크 장비 연결 구성을 데이터베이스에 저장하고, 오픈 소스 기반 네트워크 모니터링 플랫폼을 활용하여 각 장비의 연결 구성 시각화에 대한 연구를 진행 할 예정이다.

.

참 고 문 헌

[1] J.-S. Kim, H.-R Oh, “Efficient Method of Collecting Network Traces for Generating Network Topology”, Journal of KIISE, Vol. 45, No. 12, pp. 1319-1328, Dec. 2018.

[2] B. Donnet, T. Friedman. "Internet Topology Discovery: a Survey." IEEE Communications Surveys & Tutorials. Vol. 9, No. 4, pp, 56-69, 2017

[3] Ochoa Aday, Leonardo, Cristina Cervelló Pastor, and Adriana Fernández Fernández. "Current trends of topology discovery in OpenFlow-based software defined networks." (2015).

[4] Pandey, Suman, et al. "Ip network topology discovery using snmp." 2009 International Conference on Information Networking. IEEE, 2009.

[5] Lowekamp, Bruce, David O'Hallaron, and Thomas Gross. "Topology discovery for large ethernet networks." ACM SIGCOMM Computer Communication Review 31.4 (2001): 237-248.

사물인터넷의 무선 네트워킹을 위한 IEEE 802.11ah(HaLow) Dongle 개발

김민철, 김영탁*
영남대학교 대학원 정보통신공학과

kmc724@ynu.ac.kr, *ytkim@yu.ac.kr

Design and Implementation of IEEE 802.11ah (HaLow) Dongle for IoT Wireless Networking

Min-Cheol Kim, Young-Tak Kim*
Dept. of Information & Communication, Yeungnam University.

요 약

IEEE 802.11ah (WiFi-HaLow) 기반 사물인터넷 구축에서는 다양한 단말 장치(사물)에 네트워킹 기능을 쉽게 추가할 수 있어야 하며, 반경 1 Km 이내 구간의 다양한 위치에 설치되는 최대 8000 개 정도의 단말 장치들을 IoT 게이트웨이/AP (access point)를 중심으로 무선 네트워크를 구성할 수 있어야 한다. 각 IoT 단말 장치들은 다양한 위치에 설치되므로 단말장치와 AP 간의 전송 채널 상태에 따라 사용가능 한 전송 속도가 달라질 수 있고 요구되는 데이터 전송 용량도 달라 질 수 있으므로 필요에 따라 송신 전력과 전송 속도를 자동 조절할 수 있어야 한다. 본 논문에서는 사물인터넷의 무선 네트워킹을 Raspberry Pi 등의 임베디드 모듈에서도 쉽게 구성할 수 있도록 USB (universal serial interface) 접속 기능을 가지는 IEEE 802.11ah/Sub-1GHz 기반의 HaLow Dongle 을 설계 및 구현하였다. 본 논문에서 개발된 HaLow Dongle 은 USB 접속을 사용하여 전력공급과 데이터 전송을 함께 처리할 수 있어 설치 및 교체가 용이하며, CC1352R Sub-1GHz RF 소자를 사용하여 100m ~ 1Km 거리에서 200Kbps ~ 1Mbps 급 전송 속도를 제공한다. 아울러, 사물인터넷 응용 프로그램을 쉽게 개발할 수 있도록 소켓 프로그래밍 API 를 함께 제공하며, 무선 통신 채널의 상태를 쉽게 파악하며, 전송 채널과 전송 속도 및 전송 전력을 쉽게 설정할 수 있고, 데이터 송수신을 소켓 프로그래밍 방식으로 쉽게 구현할 수 있게 하였다. 본 논문에서는 개발된 HaLow Dongle 의 기능 구조와 소프트웨어 모듈에 대하여 설명하고, 성능 측정 결과를 분석한다.

Ⅰ. 서론

사물인터넷 구축에서는 반경 1Km 정도의 영역에 분산되어 있는 최대 8000개 정도의 사물인터넷 단말장치들을 IEEE 802.11ah (WiFi-HaLow) 표준 기반의 무선 통신 채널을 사용하여 IoT 게이트웨이/AP(access point)를 중심으로 무선 네트워크를 구성하여야 한다. 사물인터넷에 연결되는 단말장치들은 다양한 위치에 설치될 수 있기 때문에 단말 장치와 IoT Gateway/AP간의 무선 전송 채널 들이 다양한 상태에 있을 수 있으므로 AP에서는 매우 효율적인 채널 제어 기능이 제공되어야 한다[1-4]. 아울러 사물인터넷 단말장치들은 대부분 간단한 임베디드 플랫폼으로 구현되므로 IEEE 802.11ah 프로토콜은 사용하는 통신모듈을 쉽게 설치할 수 있게 하고, 필요한 경우 교체가 용이하게 하여야 한다.

사물인터넷 단말장치는 제한된 배터리 용량으로 구동되기 때문에 에너지 소모를 최소화할 수 있어야 하며 넓은 지역에 걸쳐 단말장치들이 분산되어 있어 다양한 주파수 간섭이 발생할 수 있어 AP는 주파수 간섭 발생을 신속하게 파악하고 채널을 변경할 수 있어야 한다. 특히, 사물인터넷 통신망에서 사용하는 900MHz 대역의 IEEE 802.11ah (WiFi-HaLow)에서는 데이터 전송 속도가 50Kbps ~ 4Mbps 으로 제한되므로, 이를 효율적으로 사용할 수 있는 MAC 프로토콜이 구현되어야 하며, 통신 채널의 상태에 따라 채널 변경과 전송 속도 및 송신 전력을 자동 조절할 수 있는 기능이 함께 구현되어야 한다. [5-9]

사물인터넷 응용 프로그램 개발에서는 기존 소켓 통신 API (application programming interface)를 사용할 수 있으면 무선 통신 기능 구현 및 네트워킹 구성에 대한 부담없이 다양한 응용 서비스를 쉽게 개발할 수 있게 된다. 사물인터넷 구축을 위하여 현재 상용제품으로 판매되는 LoRaWAN의 경우 전송거리는 최대 20Km로 우수하나 전송 속도가 290bps ~ 50Kbps로 제한적이며, 소켓 API를 제공하지 않고, 무선 채널을 자동적으로 조절할 수 있는 프로그램 환경을 제공하지 않아 사용이 제한적이다. [10-12] 특히, LoRaWan은 IEEE 802.11ah (WiFi-HaLow) 표준을 따르지 않고 독자적인 무선 전송 기술을 사용하고 있다.

본 논문에서는 사물인터넷의 Sub-1GHz 무선 네트워킹을 Raspberry Pi 등의 임베디드 모듈에서 구성하여 다양한 사물 인터넷 응용 서비스를 구현 및 시험할 수 있도록 USB (universal serial interface) 접속 기능을 가지는 HaLow Dongle을 설계 및 구현하였다. 본 논문에서 개발된 HaLow Dongle은 USB 접속을 사용하여 전력공급과 데이터 전송을 함께 처리할 수 있어 설치 및 교체가 용이하며, CC1352R Sub-1GHz RF소자를 사용하여 100m ~ 1Km 거리에서 200Kbps ~ 1Mbps급 전송속도를 제공한다. 아울러, 사물인터넷 응용 프로그램을 쉽게 개발할 수 있도록 소켓 프로그래밍 API를 함께 제공하여 사물인터넷 데이터 송수신을 쉽게 구현할 수 있게 하였으며, 네트워크 소프트웨어 모듈에서 많이 사용되는 ioctl (input output control) API를 사용하여 무선 통신 채널의 상태를 쉽게 파악하며, 전송 채널과 전송 속도 및 송신 전력을 쉽게 설정할 수 있게 하였다. 본 논문에서는 개발된 HaLow Dongle의 기능 구조와 소프트웨어 모듈에 대하여 설명하고, 채널의 변경 및 전송 속도와 송신 전력의 자동 조절 기능에 대하여 설명한다. 아울러, 실제 반경 1Km 영역에서 사물인터넷 데이터 전송 성능을 측정하고 분석하였다.

본 논문의 구성은 다음과 같다. II절에서는 사물 인터넷의 무선 통신 네트워킹 관련 연구를 간략히 소개한다. III절에서는 본 논문에서 제안하는 HaLow Dongle의 기능 구조와 소프트웨어 모듈에 대하여 설명한다. IV절에서는 구현 및 실험결과에 대하여 설명하며, V절에서 결론과 함께 향후 연구 개발을 소개한다.

Ⅱ. 관련 연구

2.1 사물인터넷 구축을 위한 무선 네트워킹 기술

사물인터넷 통신을 위하여 Wi-Fi Alliance가 Sub-1GHz 대역의 IEEE 802.11ah(Wi-Fi HaLow)에서는 CSMA/CA를 통한 충돌회피 방식을 표준화하였으며[4], 대부분의 국가들에서 900MHz 주파수 대역을 할당하고 있다. 한국에서는 무선설비규칙에서 RFID/USN 등의 무선 설비를 위하여 917 ~ 923.5MHz 주파수 대역에 200 KHz 대역폭의 32개 채널을 사용하도록 규정하고 있다. [5] 이 주파수 대역에서의 채널 최대 송신 전력은 채널마다 정리하고 있으며, 채널 2, 5, 8, 11, 14 및 17에서는 최대 4W 이하, 채널20 ~ 32에서는 200mW이하로 규정하고 있다[5].

IEEE 802.11ah에서 표준화하고 있는 CSMA/CA 방식의 채널 접속 제어 방식은 8000개 규모로 많은 단말장치가 접속되는 중앙집중식 대규모 M2M환경에서는 성능이 저하되는 것으로 분석되어 CSMA/CA와 TDMA 방식을 혼용하는 Hybrid CSMA/CA-TDMA 접속제어 방식이 제안되고 있다 [6].

2.2 사물인터넷 구축을 위한 상용 무선 네트워크 모듈

(1) LoRaWAN

사물인터넷 통신망 구축에서 사용할 수 있는 네트워크 모듈로는 LoRaWAN [10-12]모듈이 있으며, 유럽 지역의 사물 인터넷 망 구축에 많이 사용되었고, 국내에서는 SK Telecom의 사물인터넷 구축에 사용되었다. LoRaWAN은 최대 20Km까지의 전송 거리를 제공하나 전송 속도는 290bps ~ 50Kbps로 제한적이며, IEEE 802.11ah 표준을 따르지 않고, 독자적인 전송 프로토콜을 사용한다.

LoRaWAN 기반의 사물인터넷 구축에 사용될 수 있는 모듈로는 LoRa-to-USB 모듈 (LoryNet – uLory) 또는 LoRa-to-Serial 모듈 (LoryNet – sLory)를 사용하여 단말장치를 구성할 수 있고, LoryGate 모듈을 사용하여 게이트웨이를 구축할 수 있다 [12].

(2) Texas Instrument CC1312R/CC1352R

Texas Instrument (TI)에서는 Sub-1GHz 대역의 무선 통신을 위하여 CC1312R 및 CC1352R (Bluetooth 기능 포함) 모듈을 상용 제품으로 제공하고 있다[13, 14]. CC1312R/CC1352R 모듈은 100m ~ 1Km 거리에서 200Kbps ~ 4Mbps의 전송 속도를 제공할 수 있으며, 전송 채널 변경 및 전송 속도 변경과 송신 출력 조정을 소프트웨어로 쉽게 구현할 수 있다.

본 논문에서는 TI CC1352R을 사용하여 IEEE 802.11ah (WiFi-HaLow) 프로토콜을 구현하였으며, FT4222 SPI/USB 접속 기능을 사용하여 USB 2.0 표준으로 컴퓨터 장치에 접속될 수 있게 구현하였다. TI CC1352R는 전송채널, 전송채널 대역폭 및 전송 속도, 송신 전력을 소프트웨어로 설정할 수 있으며, listen-before-talk 기능과 frame error check 기능을 firmware 기능으로 제공하므로 CSMA/CA 기능을 쉽게 구현할 수 있다. 하지만 IEEE 802.11ah MAC (medium access control) 프로토콜은 별도로 구현하여야 하며, 무선 채널의 설정 (채널 중심 주파수, 채널 대역폭 및 전송 속도, 송신 출력 등) 기능은 추가적으로 구현하여야 한다.

(3) NewRaCom NRC7292

NewRaCom의 NRC7292 모듈은 IEEE 802.11ah draft 8.0 표준을 만족하며, 750 ~ 950 MHz 주파수 영역을 사용하며 150Kbps ~ 15Mbps 전송속도를 제공하는 것으로 발표되었으며, 현재 evaluation kit을 공개하고 있다 [15].

Ⅲ. IEEE 802.11ah/Sub-1GHz 사물인터넷 무선 네트워킹을 위한 HaLow Dongle

3.1 IEEE 802.11ah/Sub-1GHz HaLow Dongle의 기능 구조

본 논문에서 제안하는 IEEE 802.11ah/Sub-1GHz HaLow Dongle의 기능 구조는 그림 1에서 보는 것과 같다. sub-1GHz RF 송신은 TI CC1352R 소자를 기반으로 구현되어 있으며, TI-RTOS 운영체제를 사용하여 IEEE 802.11ah 프로토콜의 기본 기능을 구현하였다.

CC1352R 기반의 무선 전송 모듈은 SPI (serial peripheral interface)를 사용하여 외부 임베디드 시스템으로 접속할 수 있는데 본 연구에서 개발된 HaLow Dongle에서는 임베디드 시스템에 쉽게 설치 및 교체를 할 수 있도록 위하여 FT4222H 소자를 사용하여 USB 접속 기능을 함께 구현하였다.

HaLow Dongle을 사용하는 사물인터넷 단말장치에서 다양한응용 서비스 소프트웨어를 쉽게 구현할 수 있도록 raw socket API를 제공하도록 socket_halow 라이브러리를 제공하며, 이는 USB1352P1 LIB와 LibFT4222 라이브러리를 기반으로 구현되어 있다. socket_halow 라이브러리는 Linux이외의 운영체제 환경에서도 쉽게 설치할 수 있도록 운영체제내부가 아닌 사용자 영역에서 구현되어 있으며, Raspberry Pi의 wiringPi와 같은 개념으로 사용할 수 있도록 구현되어 있다.

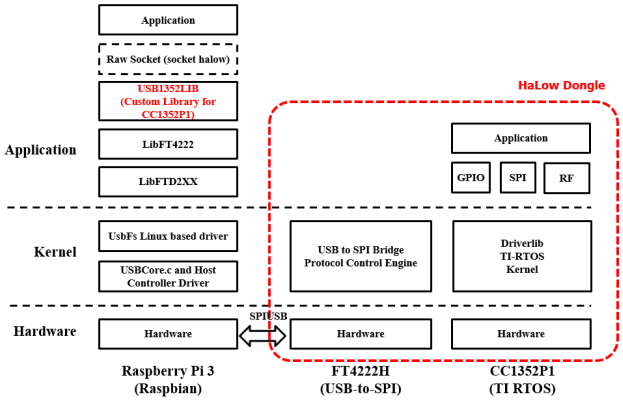


그림 1. IEEE 802.11ah/Sub-1GHz HaLow Dongle 기능 구조

3.2 HaLow Dongle의 Socket_Halow API

본 연구에서 개발되는 HaLow Dongle에서는 sub-1GHz RF 무선 통신 채널의 전송 전력과 전송 속도, 그리고 중심 주파수를 소프트웨어 기반으로 쉽게 재설정하고, 일반적인 인터넷 응용 프로그램에서 많이 사용되는 소켓 API를 제공한다. 표 1은 socket_halow API에서 제공하는 라이브러리 함수들을 설명한다.

아울러, 네트워크 소프트웨어 모듈에서 많이 사용되는 ioctl (input output control) API를 사용하여 무선 통신 채널의 파라미터를 제어할 뿐만 아니라, 무선 통신 채널의 상태를 파악하고, PER (Packet Error Rate)와 RSSI (Receive Signal Strength Indication) 측정 결과를 파악할 수 있게 한다. 표 2는 socket_halow API에서 제공하는 ioctl 명령어들을 설명하고 있다.

만약 무선 통신 채널에 장애가 생겨 deadlock이 발생할 경우 무선 통신 모듈의 디바이스 power reset 기능을 소프트웨어로 제어할 수 있도록 한다. 이러한 무선 네트워크 모듈 제어 API를 사용자들에게 제공함으로써 무선 통신 채널의 상태와 파라미터의 설정 기능을 쉽게 구현할 수 있

게 하였으며, 무선 통신 중에 문제가 생겼을 경우 디바이스 power reset 기능을 사용하여 통신 장애가 발생했을 경우 자동 복구 기능을 구현할 수 있게 하였다.

표 1. Socket_HaLow API

Socket API	설명
socket(int domain, int type, int protocol)	소켓 생성
sendmsg(socket, msg, size)	메시지 송신
recvmsg(socket, msg, size)	메시지 수신
ioctl(socket, cmd, arg)	ioctl을 이용하여 모듈에게 명령 (RF 채널 변경, data rate 변경, RF 채널 상태 확인 등)
probe(socket)	Ioctl (bind) 보다 먼저 실행 모듈과 호스트의 연결 상태 확인
bind(socket, addr, addr_length)	소켓의 주소를 바인딩
setsockopt(sock, opt_nm, opt_val, opt_size)	소켓의 옵션을 설정
getsockopt(sock, opt_nm, opt_val, opt_size)	소켓의 옵션을 읽음

표 2. Socket_HaLow API 의 IOCTL 명령어

IOCTL CMD	Comment
HALOW_IOC_RESET	디바이스 리셋
HALOW_IOC_RD_STATUS	채널 상태 (RSSI, PER, Throughput) 확인
HALOW_IOC_WR_DATA_RATE	Data rate 설정
HALOW_IOC_WR_TX_POWER	Tx Power 설정
HALOW_IOC_WR_CENTER_FREQ	중심 주파수 설정
HALOW_IOC_WR_ALL_PARAMS	전체 파라미터 (Data rate, Tx Power, Center freq) 설정
HALOW_IOC_RD_DATA_RATE	현재 Data rate 읽기
HALOW_IOC_RD_TX_POWER	현재 Tx Power 읽기
HALOW_IOC_RD_CENTER_FREQ	현재 중심 주파수 읽기
HALOW_IOC_RD_ALL_PARAMS	현재의 전체 파라미터 (Data rate, Tx Power, Center freq) 읽기
HALOW_IOC_RD_RSSI	누적된 평균 RSSI 확인
HALOW_IOC_RD_PER	누적된 평균 PER (Packet Error rate) 확인

3.3 SPI 프레임 구조

Texas Instrument (TI)에서 제공하는 Sub-1GHz 대역의 무선 통신 MCU인 CC1352P1은 외부와의 연결을 위한 SPI, I2C, I2S, UART, JTAG을 지원하는데, HaLow Dongle에서는 이들 중, 가장 빠른 속도를 지원하는 SPI를 선택하고 이를 Future Technology Device International (FTDI)에서 제공하는 USB-to-SPI Bridge IC로 연결하여 USB 기반의 SPI를 사용할 수 있도록 구성하였으며, 호스트는 FTDI의 LibFT4222 라이브러리를 사용하여 USB를 SPI처럼 사용할 수 있도록 하였다.[16, 17] 그리고 Raspberry Pi의 wiringPi와 같은 개념으로 USB1352LIB 이라는 이름으로 CC1352P1과 호스트간의 통신을 위한 함수들을 제공하는 라이브러리를 새롭게 작성하였다.

USB1352LIB는 CC1352P1와 호스트간의 데이터 송/수신과 통신 제어 및 관리 기능을 제공하는 SPI 프레임 구조와 프로토콜을 설계하였으며, LibFT4222H API를 이용하여 데이터 또는 제어 명령들을 USB 신호로 변환하여 FT4222H에게 전송하고, FT4222H는 이 제어 명령을 SPI 프레임으로 변환하여 CC1352P1에게 전송하여 호스트가 USB를 이용하여 HaLow Dongle을 직접 제어할 수 있도록 구현하였다.

그림 2는 호스트가 HaLow Dongle을 제어하기 위한 SPI 프레임의 구조를 나타낸다. SPI 프레임의 구조는 Sync Byte, FC (Frame Control), RFC (Receive Frame Control), Length, Payload로 이루어져 있다. Sync Byte는 모듈과 호스트간의 통신의 동기를 맞추며, FC (frame control)는 HaLow Dongle을 제어하기 위한 프레임 종류 (frame type), 명령어 종류 (command type) 및 세부 명령어 종류 (command sub-type)을 설정한다. RFC (RF control) 필드는 CC1352R RF 모듈의 Queue의 크기와 RF영역에서 수신 받은 패킷의 RSSI값을 저장하고 있으며, Length는 Payload의 크기이고, Payload는 전송하고자 하는 데이터를 저장한다.

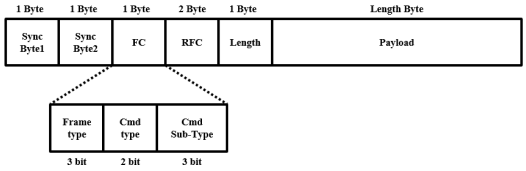


그림 2. SPI 프레임의 구조

표 3. SPI 프레임 종류 및 기능

Frame type			Command type		Command sub type			Comment
0	0	1	0	1	0	0	1	일반 데이터 전송
0	1	0	0	1	0	0	1	디바이스 리셋
0	1	0	0	1	0	1	0	RF Tx 버퍼 사이즈 확인
0	1	0	1	0	0	0	1	Tx 파워 설정
0	1	0	1	0	0	1	0	중심 주파수 설정
0	1	0	1	0	0	1	1	Data rate 설정
0	1	0	1	0	1	0	0	전체 파라미터 설정
0	1	0	1	1	0	0	1	RSSI 값 확인
0	1	0	1	1	0	1	0	Throughput 확인
0	1	0	1	1	0	1	1	Packet Error Rate 확인
0	1	0	1	1	1	0	0	현재 설정 data rate 확인
0	1	0	1	1	1	0	1	현재 설정 Tx 파워 확인
0	1	0	1	1	1	1	0	현재 설정 중심 주파수 확인
0	1	0	1	1	1	1	1	현재 설정 전체 파라미터 확인
1	0	1	0	1	0	0	1	Rx 일반 데이터
1	0	1	1	0	0	0	1	전달받은 Tx 파워 설정 값
1	0	1	1	0	0	1	0	전달받은 중심 주파수 설정 값
1	0	1	1	0	0	1	1	전달받은 data rate 설정 값
1	0	1	1	0	1	0	0	전달받은 전체 파라미터 설정 값
1	0	1	1	1	0	0	1	현재 RSSI 값

3.4 RF 프레임 구조

본 연구에서는 sub-1GHz RF 통신을 위해 그림 3과 4에서 나타낸 3개의 프레임 (U-Frame, S-Frame, I-Frame)을 정의하였다. U-Frame은 무선 통신을 시작하기 위한 beacon frame과 beacon에 대한 Ack, 그리고 여러 데이터를 Block 단위로 나누어 전송하기 전, 상대 디바이스와의 handshaking 절차에 사용할 프레임이 정의되어 있다. 그리고, 무선 통신 파라미터를 설정할 프레임과 상대 디바이스를 제어하기 위한 프레임이다.

U-Frame의 Payload를 통해 변경된 RF 파라미터를 상대 디바이스에게 알려줄 수 있도록 RF 파라미터를 Payload에 정의한다.

I-Frame은 RF를 통해 전송하고자 하는 데이터 프레임이며, Flow control을 위한 Tx 시퀀스 번호와 Rx 시퀀스 번호와 디바이스의 주소와 상대 디바이스의 주소 그리고 Payload의 길이를 포함하고 있으며, Payload에는 전송하고자 하는 데이터가 포함되어 있다.

S-Frame은 I-Frame에 대한 Ack 프레임이며 S-Frame의 Rx 시퀀스 번호를 통해 상대 디바이스로부터 수신한 I-Frame이 손실이 일어났는지 확인할 수 있으며 이를 통해 Flow control이 가능하다. I-Frame을 수신할 준비가 되었다면 S_Code를 00 (Receive Ready)상태를 호스트에게 알려주게 되고, 그렇지 않다면 S_Code를 10 (Receive Not Ready)상태를 호스트에게 알려주게 된다. S_Code가 Receive Ready인 S_Frame을 수하게 된다면 다음 데이터를 전송하게 된다. 그렇지 않고 S_Code가 10 (Receive Not Ready)를 수신하게 되면 일정 시간을 대기한 다음, RF 데이터 큐에다가 수신한 데이터를 insert한다. 이를 통해 Flow control이 가능하다.

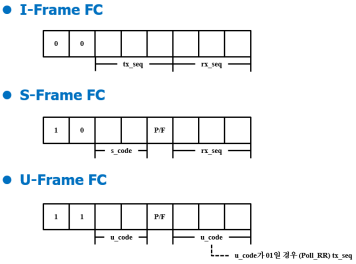


그림 3. RF 프레임의 헤더 구조

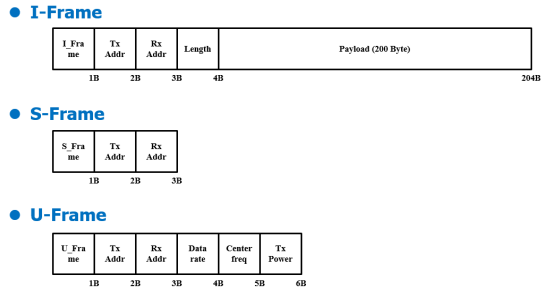


그림 4. RF 프레임 구조

표 4. U Frame의 종류 및 기능

U_code_1		U_code_2			Comment
0	0	0	0	0	Beacon
0	0	0	0	1	
0	0	1	0	0	Un numbered Poll (up)
0	0	1	1	0	U_Ack
0	1	0	0	0	Poll_RR
0	1	0	0	1	Block Transfer Request Ack (BTXRA)
0	1	0	1	0	Block Transfer Request (BTXR)
0	1	0	1	1	Block Transfer Ack (BTA)
1	0	0	0	1	Channel Re_Config_Req (CHRC_Req)
1	0	0	1	0	Channel Re_Config_Ack (CHRC_Ack)
1	1	0	0	1	Reset
1	1	0	1	1	Set Normal Response Mode (SNRM)
1	1	1	0	0	Set Asynchronous Balanced Mode (SABM)
1	1	1	0	1	Exchange ID (EXID)

표 5. S Frame의 종류 및 기능

S_code		Comment
0	0	Receiver Ready (RR)
0	1	Reject (REJ)
1	0	Receive Not Ready (RNR)
1	1	Selective Reject (SREJ)

3.5 IoT 데이터 전송에서의 흐름제어 (flow control)과 오류제어 (error control)

호스트와 CC1352P1 사이는 SPI 로 이루어져 최대 2Mbps 의 데이터 속도를 사용할 수 있는 반면에 디바이스 간의 속도는 최대 1Mbps 로 제한되어 있다. 따라서 본 연구에서는 SPI 를 통해 RF 데이터 큐의 크기를 SPI 프레임의 RFC 영역에 저장하여 이를 호스트에게 전송한다. 호스트는 수신한 SPI 프레임의 RFC 의 큐 사이즈를 수신 받을 때 마다 확인하여 큐 사이즈가 일정 값 이상이 되면 1ms 대기하고 dummy frame 을 전송하여 RF 데이터 큐의 크기를 확인하고, 확인된 RF 데이터 큐의 크기가 일정 값 미만이면 데이터를 SPI 를 통해 전송하고, 그렇지 않다면 다시 1ms 대기하고 dummy frame 을 전송하여 RF 데이터 큐의 크기를 확인한다. 이를 통해 데이터의 전송량을 조절하여 SPI ↔ RF 간의 데이터 손실이 발생하지 않도록 구현하였다.

HaLow Dongle 은 무선 통신 모듈이기 때문에 데이터의 손실이 발생할 수 있다. 따라서 I-Frame 을 전송할 때, Tx 시퀀스 번호와 Rx 시퀀스 번호를 piggyback 방식으로 I-Frame 에 표시하고, 이를 수신단에서 기록한 Tx 시퀀스 번호와 Rx 시퀀스 번호를 매칭하여 시퀀스 번호가 올바른 경우, Ack 프레임을 전송하여 올바른 데이터가 전송됐다는 것을 송신단에게 알려주고, 그렇지 않다면 NAck 프레임을 전송하여 올바르지 않은 데이터가 전송됐다는 것을 송신단에게 알려준다. 송신단은 Ack 프레임을 통해 다음 데이터를 전송을 하게 되고, NAck 프레임을 수신하거나 Ack 를 수신하지 못했다면 손실된 데이터를 재전송 하도록 구현하였다.

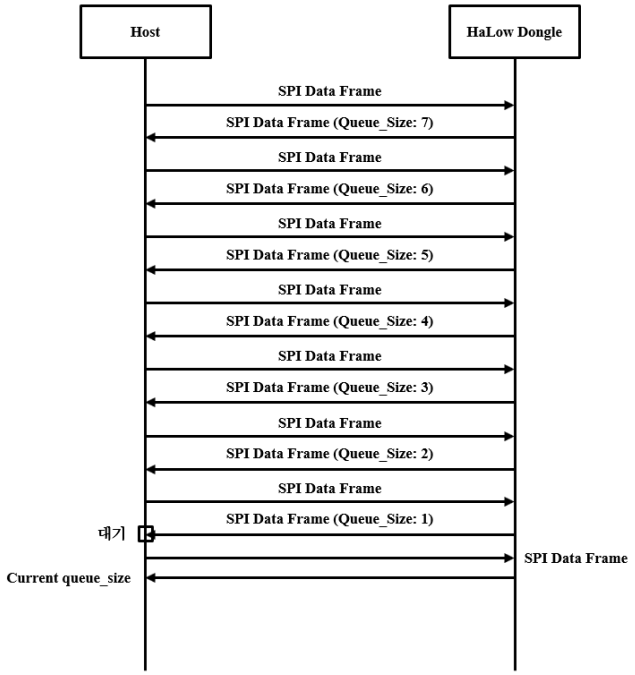


그림 5. SPI Flow Control Sequence diagram

3.6 기능 순서도 (sequence diagram)

HaLow Dongle 을 사용한 데이터 전송은 2 단계를 거쳐서 이루어진다. 먼저 HaLow Dongle 을 초기화하고 무선 통신 채널의 초기화(default) 설정이 이루어 진 다음, 데이터를 전송할 수 있는 단계에 도달한다. HaLow Dongle 초기화 단계에서는 AP 로부터 각각의 디바이스가 U Frame 의 Beacon Frame 을 수신 받고, 그 시점부터 디바이스의 RF 스레드가 동작하여 AP 와 다른 디바이스 간의 동기를 맞추고, 이에 대한 Ack 를 전송하여 AP 가 디바이스가 얼마나 있는지 파악할 수 있도록 구현하였다. 그리고 무선 통신 채널의 초기화(default) 설정 단계에서는, 속도가 가장 높고 전력 소모가 적은 설정으로 초기화된 설정 값을 AP 가 각 디바이스들에게 전송하고 수신한 각 디바이스들이 이러한 상태로 처음 채널을 설정하게 된다. 위 단계를 거치게 되면 HaLow Dongle 은 데이터를 송/수신할 준비를 마치고 IoT 센서 및 액추에이터로부터 수신한 데이터를 전송할 수 있는 단계에 도달하게 된다. 이 단계에서는 I-Frame 을 전송하면 이에 대한 Ack 를 수신하는 Stop & Wait 방식을 사용하며, 채널의 용량을 최대한 사용할 수 있도록 piggyback 기능을 추가하여 I Frame 을 통해 데이터를 수신하고 전송할 데이터가 데이터 큐에 존재할 경우, 이를 Ack 대신 전송하여 Ack 의 역할을 I Frame 이 대신 수행할 수 있도록 구현하였다.

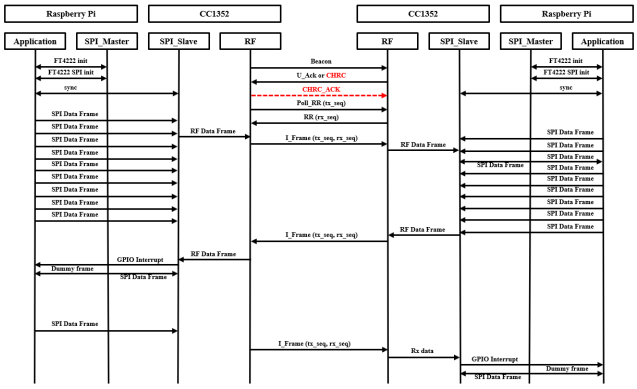


그림 6. 전체 시퀀스 다이어그램

IV. HaLow Dongle 구현 및 실험결과

4.1 HaLow Dongle의 Sub-1GHz 무선 통신 모듈 구현

본 연구에서 개발한 HaLow Dongle 무선 통신 모듈은 TI(Texas Instruments)사의 Sub-1GHz, 2.4GHz Dual band 저전력 RF Microcontroller인 CC1352P1을 기반으로, PA(Power Amplifier)와 LNA(Low noise amplifier)기능을 GPIO로 조절하여 2.4GHz 대역의 송신 전력을 낮추고 Sub-1GHz 대역의 송신 전력을 높여 보조실험을 하도록 구성하였다. 그림 7은 본 논문에서 구현한 HaLow Dongle USB 무선 통신 모듈의 기능 블록도를 보여준다. TI CC1352P1은 5kbps (Long Range Mode) ~ 2Mbps (High Speed Mode)까지 다양한 전송속도와 14dBm ~ 20dBm까지의 전송 전력 및 917MHz ~ 923MHz까지의 중심 주파수를 제공한다.

그림 7의 구조에서 CC1352P1 RF 모듈은 Sub-1GHz 무선 채널을 사용한 프레임 송신 및 수신 기능을 담당하며, Listen-before-Talk 기능을 구성할 수 있다. 실제 사물 인터넷 응용프로그램 기능은 Raspberry Pi 모듈에 설치된 Raspbian 리눅스 운영체제와 3.2절에서 설명한 소켓 API에 기반한 HaLow Dongle용 Socket API를 사용하여 USB1352LIB과 연결하고, 사물인터넷 무선 통신 네트워크 응용 프로그램을 쉽게 구현할 수 있도록 한다.

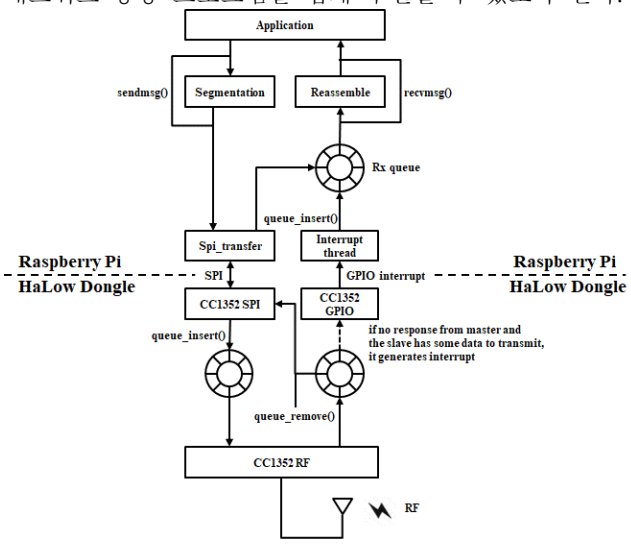


그림 7. HaLow Dongle USB 무선 통신 모듈 블록 다이어그램

4.2 HaLow Dongle USB 무선 통신 모듈의 데이터 프레임 전달 성능 분석

HaLow Dongle의 무선 통신 모듈은 에너지 효율적인 전송 기능을 갖추기 위하여 전송 거리, 전송 속도 별 비트에러 발생률을 측정하고 분석하였으며, 프레임에 비트에러가 발생하지 않는 범위에서 송신 전력을 최소한으로 사용할 수 있도록 하였다.

먼저 50Kbps ~ 1Mbps 전송속도에서 실제 데이터 전송속도를 측정하였다. 그림 8은 AP가 노드에게 할당하는 TDMA Slot 크기를 고정하고, 전송속도를 가변 함에 따라서 증가하는 throughput을 양방향과 단방향으로 측정하고 분석한 결과이다. 전송속도가 1Mbps로 설정된 경우, 양방향 전송에서는 746.38Kbps의 throughput이 제공 가능하며, 단방향 전송에서는 566.15Kbps의 throughput이 가능하다. 양방향 전송의 throughput이 더 높은 이유는 piggyback 기능을 사용하여 반이중 (half-duplex) 전송 방식에서의 ACK 프레임 전송 부담을 없애도록 구현되어 있기 때문이다.

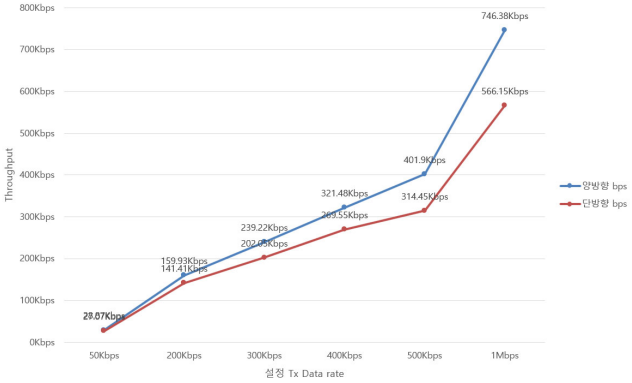


그림 8. 전송 속도 별 Throughput

HaLow Dongle 의 각 속도 별로 통신 가능 최소 RSSI 를 측정하였다. 표 6 은 AP 가 전송속도를 가변함에 따라 HaLow Dongle 이 수신하는 최소 RSSI 값을 측정하였다. 이는 HaLow Dongle 로 통신 가능한 최소 RSSI 이며, 측정 값보다 높을 경우 통신이 두절되며, RSSI 가 측정 값보다 낮아도 패킷 손실이 발생할 수 있다.

표 6. 각 속도 별 통신 가능한 최소 RSSI

data rate	최소 RSSI
50kbps	-95dBm
200kbps	-92dBm
300kbps	-88dBm
400kbps	-88dBm
500kbps	-80dBm
1Mbps	-75dBm

그림 9 ~ 10은 AP가 전송속도를 가변함에 따라 HaLow Dongle이 RSSI의 레벨에 따른 Throughput과 Packet Error Rate (PER)을 양방향 통신과 단방향 통신으로 나누어 측정한 결과값을 비교하여 보여준다.

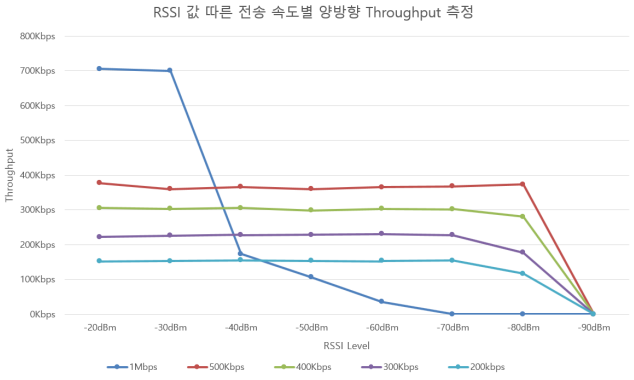


그림 9. RSSI 값에 따른 전송 속도별 양방향 Throughput 측정

전송속도가 200~500Kbps인 경우 -80dBm이내의 RSSI 수신 신호 세기에서는 정상적인 throughput이 가능하나 -90dBm 이하에서는 데이터 전송 성능이 급격하게 감소하는 것이 측정되었다. 특히, 전송속도가 1Mbps인 경우 RSSI값이 -30dBm 이하에서는 전송 성능이 감소하기 시작하고, -60dBm이하에서는 급격하게 감소하여 추가적인 분석과 최적화가 필요한 것으로 파악된다.

그림 10은 RSSI 값에 따른 전송 속도별 양방향 PER (packet error rate) 측정값을 비교하여 보여준다. 전송속도가 200Kbps ~ 500Kbps인 경우 -70dBm 이하에서는 패킷 에러율이 증가하기 시작하며, -90dBm이하에서는 심각한 패킷 에러가 발생하는 것을 알 수 있다. 전송 속도가 1Mbps인 경우 RSSI가 -60dBm 이하에서 패킷 에러율이 급격하게 증가하는 것으로 측정되었다.

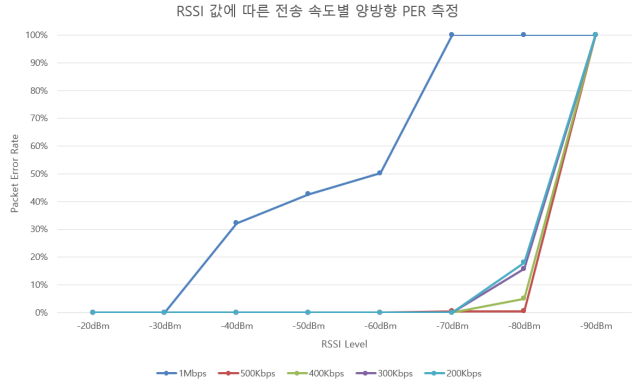


그림 10. RSSI 값에 따른 전송 속도별 양방향 PER 측정

V. 결론

본 논문에서는 IEEE 802.11ah 기반 사물인터넷의 Sub-1GHz 무선 네트워킹을 다양한 임베디드 모듈에서 쉽게 구성하여 다양한 사물 인터넷 응용 서비스를 구현 및 시험할 수 있도록 USB (universal serial interface) 접속 기능을 가지는 HaLow Dongle을 설계 및 구현하였다. 본 논문에서 개발된 HaLow Dongle은 USB 접속을 사용하여 전력공급과 데이터 전송을 함께 처리할 수 있어 설치 및 교체가 용이하며, CC1352R Sub-1GHz RF소자를 사용하여 100m ~ 1Km 거리에서 200Kbps ~ 1Mbps급 전송 속도를 제공할 수 있다. 아울러, 사물인터넷 응용 프로그램을 쉽게 개발할 수 있도록 소켓 프로그래밍 API를 함께 제공하여 사물인터넷 데이터 송수신을 쉽게 구현할 수 있게 하였으며, 네트워크 소프트웨어 모듈에서 많이 사용되는 ioctl (input output control) API를 사용하여 무선 통신 채널의 상태를 쉽게 파악하며, 전송 채널과 전송 속도 및 송신 전력을 쉽게 재설정할 수 있게 하였다.

본 논문에서는 개발된 HaLow Dongle의 기능 구조와 소프트웨어 모듈에 대하여 설명하고, 채널의 변경 및 전송 속도와 송신 전력의 자동 조절 기능에 대하여 설명한다. 아울러, 실제 반경 1Km 영역에서 사물인터넷 데이터 전송 성능을 측정하고 분석하였다. 본 연구의 다음 계획으로는 반경 1Km 구역에 포함된 최대 8000개의 사물인터넷 단말장치들을 하나의 IoT gateway로 접속할 수 있게 하고, 이들 사물 인터넷 단말장치를 외부 인터넷에서 주어진 권한에 따라 쉽게 접근하고 제어 할 수 있도록 하며, 클라우드 서비스에 연동될 수 있도록 기능을 추가 개발하는 것을 추진하고 있다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음 (IITP-2020-2016-0-00313)

참 고 문 헌

[1] Jie Lin et.el, “A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications”, IEEE Internet of Things JOURNAL Vol. 3 NO.5, oct, 2017, pp. 1125-1142.

[2] Corrales Madueno, Cedomir Stefanovic and Popovski, “Reliable and Efficient Access for Alarm-Initiated and Regular M2M Traffic in IEEE 802.11ah Systems”, IEEE INTERNET OF THINGS JOURNAL VOL. 3 NO.5, oct, 2016, pp. 673-682.

[3] Stefan Aust, Venkatesha Prasad and Ignas G. M. M. Niemegeers, "Outdoor Long-Range WLANs: A Lesson for IEEE 802.11ah," IEEE COMMUNICATION SURVEYS & TUTORIALS, vol. 17, no. 3, 2015.

[4] T. Adame, A. Bel, B. Bellalta, J. Barcelo, and M. Oliver, “IEEE 802.11AH: The WiFi approach for M2M communications,”IEEE Wireless Communication, vol. 21, no. 6, pp. 144- 152, Dec. 2014.

[5] 무선 설비 규칙, 미래창조부 고시 2016-125 호, 2016. 11. 30.

[6] Nurullah Shahin, Rashid Ali and Young-Tak Kim., “Hybrid Slotted-CSMA/CA-TDMA for Enhanced Registration of Massive IoT Devices,”IEEE Access Vol. 6, pp 18366 - 18382, 2018.

[7] 김기태, 김영탁, “에너지 효율적인 사물 인터넷 통신을 위한 Sub-1GHz 무선 통신 채널의 전송 속도 및 송신 전력의 스마트 제어 기법”, 한국통신학회 하계 학술대회, 2017.

[8] 정용환, 김영탁, “IEEE 802.11ah 기반의 사물인터넷 통신을 위한 CSMA/CA 와 TDMA 의 하이브리드 전송 제어 기법”, 한국통신학회 하계 학술대회, 2017.

[9] 김민철, 김영탁, “IEEE 802.11ah/Sub-1GHz 기반의 사물인터넷 스마트 제어”, KNOM 2019.

[10] LoRa to USB 컨버터 (LoryNet - uLory), http://lory.co.kr/kor/LoryNet/view.php?part=1&idx=2.

[11] Andreas Spiess, “How to build a LoRa / LoraWAN Gateway and connect it to TTN?,” Youtube, https://www.youtube.com/watch?v= ZFVA6cQyheY.

[12] 시스템 베이스 쇼핑몰, http://sysbasmall.com/product/slory.

[13] Texas Instrument (TI) CC1312R SimpleLink™ High-Performance Sub-1 GHz Wireless MCU, http://www.ti.com/lit/ds/swrs210f/swrs210f.pdf.

[14] Texas Instrument (TI) CC1352R SimpleLink™ High-Performance Multi-Band Wireless MCU, http://www.ti.com/lit/ds/symlink/cc1352r.pdf.

[15] NewRaCom NRC7292 EVK, http://newracom.com/.

[16] Future Technology Device International Ltd. FT4222H (USB2.0 to QuadSPI/I^{2}C Bridge IC), https://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT4222H.pdf.

[17] Future Technology Device International Ltd. User Guide for LibFT4222 Version 1.4, https://www.ftdichip.com/Support/Documents/AppNotes/AN_329_User_Guide_for_LibFT4222.pdf.

SLO 요구 변화에 따른 기상 대기 응용의 프로파일링 적용 실험 계획법

오지선*, 김세진, 김윤희
숙명여자대학교 컴퓨터과학과

{* jsoh8088, wonder960702}@gmail.com, yulan@sookmyung.ac.kr

An Experiment Planning Method using MPAS Execution Profiles for SLO Requirement Changes

Jisun Oh* , Sejin Kim, Yoonhee Kim
Dept. of Computer Science, Sookmyung Women's University

요 약

재난, 재해와 같은 특수 상황에서의 기상 예보는 시간적 한계를 가지고 의미를 도출해야 한다. 기상 응용인 MPAS(Model for Prediction Across Scales)는 응용의 수행에 있어 선택하는 격자 및 반복 횟수에 따라 사용하는 메모리 및 수행시간이 달라진다. 그러므로 이에 대한 프로파일링 정보를 사전에 수집하면 사용자의 요구사항에 따라 적합한 격자를 선택하여 제공할 수 있다. 본 논문에서는 MPAS 를 대상으로 하여 격자 크기 및 반복 횟수에 따른 프로파일링 정보를 수집하고, 이를 활용하여 사용자의 시간적 제약사항이 변화할 때 이에 따른 적합한 실험을 계획한다. 또한, 실험을 통해 프로파일링 기반 실험 조절이 SLO(Service Level Objectives) 요구가 변화하여도 이를 충족함을 보인다.

Ⅰ. 서 론

기상 예보 응용과 같이 시간 제약 하에 결과를 도출해야하는 연구에 있어 응용 실행 관리는 중요하다. 기상청은 기상 모델의 사용에 있어 고성능 자원을 활용하여 예보를 제공하고 있다. 하지만 여전히 시간적인 한계를 가지고 의미 있는 결과를 도출해야 하는 고성능 연구 환경 제공은 여전히 어렵다. 특히 홍수, 태풍과 같은 더 빠른 예보가 필요한 특수 상황에서 한계를 보인다. 이는 실시간 분석 및 적용을 위해 필요한 응용 프로그램의 이해에 기반한 지능적 계산 배치가 적절히 이루어지지 않았기 때문이다.

기후 및 기상 예측 연구에 사용하는 대기, 해양 및 기타 지구 시스템 시뮬레이션 응용인 MPAS(Model for Prediction Across Scales)[1]는 격자 및 반복 횟수에 따라서 사용하는 메모리 및 수행시간이 달라진다. 그러므로 사용자의 시간적 요구사항이 존재하는 경우, 프로파일링 정보에 따라 적합한 격자를 선택해 제공하여 시간 제약 조건을 만족시킬 수 있다. 또한, 예보 정확도의 요구사항이 존재하면, 조밀한 간격을 통하여 정확한 예보를 제공하여 정확도 제약조건을 만족시킬 수 있다. 이는 예보 상황에 따라서 프로파일링 데이터를 활용하여 실험 조건을 제시함으로써 시간 내 예보 가능성을 높일 수 있음을 의미한다.

본 논문에서는 기상 대기 응용인 MPAS 를 대상으로 하여 프로파일링 정보를 수집하고, 이를 기반으로 하여 예보의 시급성에 따라 시간 요구사항이 변동하였을 때 실험 조건을 변경하여 제약 조건 내에 예보를 할 수 있도록 실험을 계획한다. 실험을 통해 프로파일링 기반 실행 조절이 데드라인이 변경하였을 때에 이에 따라 SLO(Service Level Objectives)를 충족함을 보여준다.

Ⅱ. 관련연구

2.1 MPAS

MPAS 응용은 기후, 지형 등 전처리 과정 단계(init_atmosphere)와 해석 단계(atmosphere)로 나뉜다. 그림 1 은 MPAS 응용 실행 구조도이다. 전처리 과정에서는 사용될 전구 도메인에 지형자료(WRF Terrestrial Data)를 생성하고, 격자 크기(Mesh Grid Data)에 따라 기상, 기후 데이터를 내삽한다. 이러한 전처리 과정이 완료되면 해석 모델을 수행하여 모델 적분을 수행한다. 그 후 나온 결과 파일을 사용하여

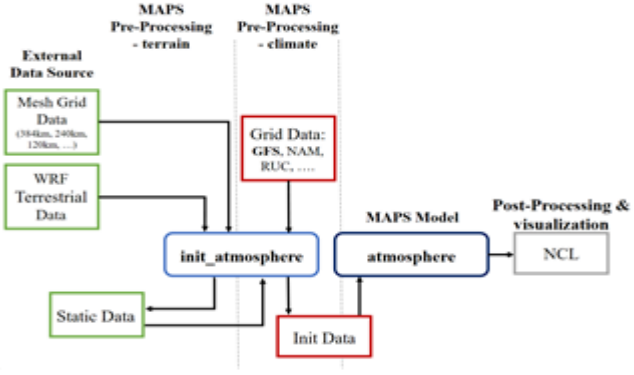


Figure 1 MPAS 실행 구조도[3]

Table 1 격자 크기와 예보 반복 횟수에 따른 수행시간, 메모리 사용량

	예보 반복 횟수	격자 크기				
		384km	240km	120km	60km	30km
수행시간	10	5m 05s	18m 37s	25m 43s	1h 38m 12s	4h 12m 44s
	5	2m 57s	9m 16s	15m 49s	48m 22s	2h 09m 07s
메모리	.	4.98GB	6.01GB	7.29GB	18.23GB	31.3GB

NCL 도구를 통해 시각화하여 예측 결과를 제공한다[2] 자세한 내용은 [3] 에서 참조하였다.

2.2 MPAS 의 자원 사용 패턴

[3] 논문은 MPAS 응용에 대해 격자의 크기에 따른 수행 시간 분석 및 메모리 조절을 통한 응용 분석을 진행하였다. 위 논문은 격자의 크기에 따른 수행 시간 측정을 위해 240km, 120km 두 격자를 사용하여 수행 시간을 측정하였다. 실제 적분 계산을 수행하는 atmosphere 해석 단계는 240km 격자를 사용하였을 때 약 18 분 33 초, 120km 격자를 사용하였을 때는 약 25 분 42 초 정도 소요되어 격자 크기가 감소함에 따라 수행시간이 약 24.73% 증가하여 조밀한 격자 크기일수록 해석 단계의 수행 시간에 차이가 나는 것을 확인하였다. 메모리 자원 조절 실험을 통해서 120KM 격자의 경우, 가용 메모리를 6GB 로 하여 실행할 때 수행 시간에 영향을 미치지 않는 최소 자원 조건임을 확인하였다. 또한, 응용의 자원 사용 패턴을 확인하기 위해 swap 및 mlock 분석을 실시하였으며 이를 통해 MPAS 응용은 mlock 으로 인한 성능 향상은 어려우며, swap out 발생이 적고 데이터 로컬리티가 좋은 것으로 예상하였다.

Ⅲ. 실험

MPAS 응용은 초기에 정의된 격자 크기 별로 사전에 도메인에 맞춰 기후, 지역 내삽 등 전처리 과정을 수행하면, 예보 시에는 해석 모델만 수행하여 예보 결과를 도출할 수 있다. 따라서 다양한 제약 조건에서 기상 예보를 수행하는 데 적합한 응용이다.

여러 제약 사항이 존재할 경우에 이를 만족하는 기상 예보 제공을 위해 프로파일링 정보 수집이 필요하다. 두 가지 조건(시간, 예보 정확도 충족)의 시나리오 실험을 위해 사전에 MPAS 응용의 격자 크기에 따른 수행 시간, 메모리 사용량과 예보 단위 따른 수행 시간에 대한 프로파일링 데이터를 수집하였다. 본 실험에서는 전지구 지형 데이터 13GB 를 사용하여 컨테이너 클러스터 환경 내에서 격자 별 이미지를 구성하였다. 기후 자료는 수십 개의 대기 및 토양 변수 데이터 세트인 GFS 를 사용하였으며. 실험 환경은 Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz, 12 cores, RAM 32GB 이며, MPAS version 은 6.2 를 사용하였다

3.1 격자 크기 별 프로파일링 데이터 수집

본 논문은 표 1 과 같이 384km, 240km, 120km, 60km, 30km 의 격자 크기와 반복 횟수 조절을 통해 프로파일링 데이터를 수집하였다. 격자 크기 별 시간 및 메모리 프로파일링 정보를 통해 사용자의 조건을 만족하는 서비스를 선택하여 제공할 수 있다. 격자 크기가 작을수록 적은 지역에 정밀한 예보를 할 수 있고, 반복 횟수가 높을 수록 예보의 정확도가 높아진다고 할 수 있다. 하지만, 격자 크기가 작을수록 결과를 도출하는데 소요되는 시간이 증가하며 반복 횟수가 많을 수록 실험에 소요되는 시간이 증가함을 알 수 있다. 표 1 에서 120km 의 경우 반복 횟수가 10 회일 때는 18 분 37 초가, 5 회일때는 9 분 16 초가 소요되었으며 메모리는 6.01GB 를 사용한다.

본 논문에서는 수집한 프로파일링 정보에 따라 시간 SLO 충족을 위해 격자 크기와 계산 반복 횟수를 조절하여 실행한다.

3.2 시간 SLO 충족을 위한 격자 크기 조절 실험

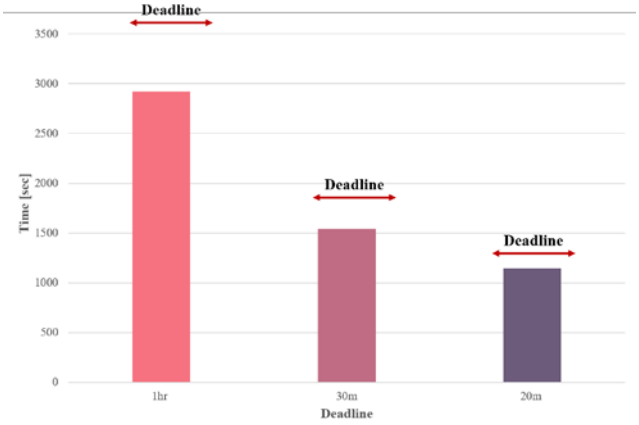


Figure 1 시간 SLO 충족을 위한 격자 크기 조절 실험 결과

시간 SLO 충족을 위한 격자 크기 조절 실험 결과는 그림 2 와 같다. 초기에 데드라인이 60 분인 경우, 격자 60km 를 사용하고 반복 횟수 5 번을 사용하여 예보하였다고 가정하자. 이는 그림 2 의 첫 번째 막대와 같이 2920 초의 시간이 소요되었다. 예보 시간이 점점 짧아져 기존 방법으로는 요구사항을 맞출 수 없어 실험의 조건을 바꿔야 할 경우에는 표 1 과 같은 프로파일링 정보를 활용하여 실험의 조건을 선택할 수 있다. 먼저, SLO 의 요구사항이 30 분으로 변경되어 120km 의 격자 크기와 반복 횟수 10 번을 선택하여 실험하였다. 이는 그림 2 의 두 번째 막대와 같이 1543 초가 소요되었다. 마지막으로 시간적 제약 조건이 20 분으로 변경 되었을 때, 240km 의 격자 크기와 반복 횟수 10 번의 실험 구성을 선택하여 그림 2 의 세 번째 막대와 같이 1144 초가 소요되어 사용자의 요구사항을 충족할 수 있었다. 이는 시간적 제약 조건이 단축되었을 때 격자 크기가 커짐에 따라 상대적으로 정밀도는 떨어질 수는 있으나 이를 보완하기 위해 반복 횟수를

증가시켰으며, 제약 조건에 맞는 실험을 진행할 수 있음을 보였다.

이를 통해 프로파일링 정보를 활용하면 사용자가 제공하는 SLO 를 중심으로 어떤 격자 및 반복 횟수의 조합을 선택할지 실험을 계획 할 수 있음을 보였다.

Ⅳ. 결론

본 논문에서는 기후 및 기상 예측 연구에 이용하는 응용인 MPAS 응용을 시간 조건에 맞춰 수행하도록 응용의 프로파일링 정보를 수집하였으며, 이를 바탕으로 하여 실행 조절 실험을 진행하였다. 실험을 통해 프로파일링 정보 기반 실행 조절은 시간 요구사항이 단축되면 격자 크기는 확장하고, 반복횟수를 늘려서 사용자의 시간 SLO 를 충족함을 확인하였다. 이는 재난 및 특수 상황으로 인해 평소보다 예보의 시간이 단축되었 때 프로파일링 정보를 통해 어떤 조건을 선택하여 실험을 진행해야 하는지 계획할 수 있음을 보여준다.

향후 연구로는 시간 및 자원적 다양한 조건 하에서 예보를 위한 클러스터 컨테이너 환경의 스케줄링 기법을 연구하고자 한다.

ACKNOWLEDGMENT

본 연구는 정부(과학기술정보통신부)의 재원으로

한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-

2015M3C4A7065646).

참 고 문 헌

[1] MPAS, https://mpas-dev.github.io/

[2] MPAS architecture,
http://www2.mmm.ucar.edu/projects/mpas/mpas_atmosphere_users_guide_6.0.pdf

[3] 오지선, 김윤희. 제약 조건에서의 예보를 위한 기상 응용의 실행 패턴분석.KNOM Review, Vol. 22, No. 3, Dec. 2019, pp. 25-30, 2019 년 12 월.

잔류물질정보 데이터베이스 자동 업데이트 시스템

신무곤, 백의준, 박지태, 김명섭
고려대학교

{tm0309, pb1069, pjj5846, tmskim}@korea.ac.kr

Automatic Database Update system for Pesticides and Veterinary Drugs Information

Mu-Gon Shin, Ui-Jun Baek, Jee-Tae Park, Myung-Sup Kim
Korea Univ.

요 약

현재 잔류물질정보를 제공하는 웹 페이지에서는 농약 및 동물용 의약품의 식품 내 잔류허용기준 정보를 제공하고 있다. 하지만 정보의 누락, 검색 시간 지연, 웹 페이지 오류 등 정보 제공이 원활하게 이루어지지 않고 있기에 사용자들이 불편을 겪고 있다. 또한 잔류허용기준이 개정 될 때마다 수동으로 정보를 업데이트 해야하는 등 불편함이 있다. 이에 본 논문에서는 잔류물질정보 제공 웹 페이지의 데이터베이스 개선방안과 잔류허용기준 개정고시 문서 및 기타 데이터의 자동 업데이트 시스템 구조에 대해 설명한다.

Ⅰ. 서 론

잔류물질정보를 제공하는 웹 페이지 에서는 농약 및 동물용 의약품의 식품 내 잔류허용기준 정보와 농약, 의약품 정보, 농약의 분석법 등 정보를 제공하고 있다[1]. 이러한 정보를 통해 국내 기업 및 농민들은 자신의 생산품의 식품 내 잔류허용기준을 측정하고 그 측정값이 국내 혹은 국제 기준을 통과할 수 있는지 가늠하는 척도로 사용하고 있다. 그러나 이 페이지를 그대로 사용하기에는 정보 누락, 검색 시간 지연, 웹 페이지 오류, 검색 불가 등 많은 문제들이 존재한다.

먼저, 농약 정보를 검색 하였을 때 전체 정보 중 약 74%의 공백이 있고 해외 자료를 제외하고는 약 60%의 정보가 누락되어 있는 것을 확인하였다. 또한 가장 중요한 정보라고 할 수 있는 국내 농약 잔류허용기준의 경우에는 약 72% 공백이 발견되어 심각한 문제를 가지고 있는 것으로 파악된다. 동물용 의약품의 경우도 마찬가지로, 많은 데이터의 누락이 있어 실제 이용자들이 효과적인 정보를 얻어가는지는 미지수이다.

그리고 정보 검색 시 많은 양의 정보를 로드 하지 않음에도 지연이 발생하며 농약 명으로 검색 시에는 농약 명이 일치함에도 관련 자료가 검색 되지 않는 등 웹 페이지 내의 오류가 빈번히 발생하고 있다. 또한 잔류허용기준이 개정될 때마다 수동으로 업데이트된 정보를 입력해야 하기 때문에 정보의 수정에 많은 어려움이 있다.

이러한 문제점들이 많이 존재하기 때문에 잔류물질정보 데이터베이스를 최신화 하고 데이터의 공백을 채우는 것이 중요하다. 본 논문에서는 데이터베이스를 최신화 하고 데이터의 공백을 채우기 위해 개정고시 문서 및 데이터 자동 업데이트 시스템을 제안한다.

본 논문은 서론에서 연구 배경과 목표를 서술하고, 본론에서 자동업데이트 시스템 구조를 제안한다.

본 연구는 2020 년도 식품의약품안전처의 연구개발비 (20162 수산물 625)로 수행된 연구임..

Ⅱ. 본론

본 장에서는 잔류물질정보 웹 페이지의 문제점을 서술하고 개정고시 문서 및 데이터 자동 업데이트 시스템 구조에 대해 설명한다.

2.1 웹 페이지의 문제점

기존의 잔류물질정보 웹 페이지의 데이터는 수동으로 업데이트 되고 있다. 잔류허용기준은 개정이 빈번하게 일어나고 한번에 많은 양이 업데이트 된다. 이를 수동으로 수정하고 입력하는 것은 많은 시간이 걸리는 작업이기 때문에 이를 자동화하는 것은 매우 중요하다.

농약 명으로 검색 시 농약 명이 일치함에도 관련 자료가 검색되지 않는 웹 페이지의 오류가 발생하고 있다. 또한 검색 시 많은 정보를 로드 하지 않음에도 지연이 길게 발생하는데, 이는 데이터베이스의 색인화가 되어 있지 않거나 웹 서버에서 데이터베이스에 요청하는 쿼리에 문제가 있는 것으로 보여진다. 이를 개선함으로써 이용자들의 편의성을 증대시킬 수 있다.

그림 1. 농약 명 검색 시 오류

	공전번호	농약명	농약이명	시약명	잔류물의정의	용도	적용대상작물	계통	IUPAC명	분자식
빈칸개수	1301	0	1069	17	1747	619	1382	1330	1134	519
공백 비율(%)	72.56	0	59.62	0.95	97.43	34.52	77.08	74.18	63.25	28.95
	상품명	구조식	분석법	분석법해설서	형태	녹는점	끓는점	증기압	LogP_ow	밀도
빈칸개수	518	833	0	1258	869	926	1781	936	1113	1778
공백 비율(%)	28.89	46.46	0	70.16	48.47	51.65	99.33	52.2	62.07	99.16
	pk_a	용해도	안전성	국내 농약잔류허용기준		ADI(한국)	Acute RfD(한국)	RESIDUE(한국)	독성요약정보(한국)	관련파일(한국)
빈칸개수	1540	860	990	1308		1267	1593	1793	1792	1418
공백 비율(%)	85.89	47.96	55.21	72.95		70.66	88.85	100	99.94	79.09
	ADI(CODEX)	Acute RfD(CODEX)	RESIDUE(CODEX)	독성요약정보(CODEX)	관련파일(CODEX)	ADI(미국)	Acute RfD(미국)	RESIDUE(미국)	독성요약정보(미국)	관련파일(미국)
빈칸개수	1548	1648	1630	1792	1616	1644	1755	1793	1792	1734
공백 비율(%)	86.34	91.91	90.91	99.94	90.13	91.69	97.88	100	99.94	96.71
	ADI(일본)	Acute RfD(일본)	RESIDUE(일본)	독성요약정보(일본)	관련파일(일본)	ADI(유럽)	Acute RfD(유럽)	RESIDUE(유럽)	독성요약정보(유럽)	관련파일(유럽)
빈칸개수	1457	1786	1793	1792	1647	1495	1613	1792	1792	1589
공백 비율(%)	81.26	99.61	100	99.94	91.86	83.38	89.96	99.94	99.94	88.62
전체공백비율	**74.44**									
해외자료 제외	**60.95**									

그림 2. 잔류물질정보 데이터 현황

그리고 농약 및 동물용 의약품 정보를 검색 하였을 때 정보의 누락이 많이 발견된다. 누락 정보에는 해외의 잔류허용기준, 약품의 특성, 분석법 등 이용자들이 많이 찾을 법한 정보가 포함되어 있다. 본 연구팀은 전체 정보 중 약 74%, 해외 자료를 제외하고는 약 60%의 정보 누락을 발견하였다.

2.2 개정고시 문서 자동 업데이트 시스템

본 절에서는 앞서 언급한 웹 페이지의 문제점 중 한가지인 데이터 수동 업데이트를 개선하기 위해 잔류허용기준 개정고시 문서 자동 업데이트 시스템을 제안한다. 제안된 시스템을 통해 관리자는 편리하게 업데이트된 정보를 데이터베이스에 업데이트 할 수 있게 되고 사용자는 업데이트 된 정보를 파악하지 못하여 피해를 보는 일이 없어질 것이다.

먼저 개정고시 문서는 hwp 파일로 작성되기 때문에 파싱하여 데이터를 추출하기가 어렵다. 따라서 hwp 문서를 html 문서로 변환하는 과정이 필요하다. html 변환을 수행하기 전에 프로그램 상에서 예외처리가 힘든 부분을 제거하는 작업이 필요하다. 변환된 html 문서에서 필요한 데이터만 추출하기 위하여 전처리 과정이 필요한데, 이 과정에서는 추출 할 데이터 이외의 태그를 제거한다.

다음 과정은 텍스트 추출 단계이다. 이 단계에서는 테이블을 제외한 키워드들을 추출한다. 추출하는 키워드들은 그림 3 과 같다.

(1) 겐타마이신(Gentamicin) : 항균제

◎ 잔류물의 정의 : Gentamicin C1a, C2, C2a 및 C1의 합을 gentamicin으로 함

그림 3. 추출 키워드

이 단계를 거쳐 추출된 텍스트 중 필요하지 않은 텍스트를 제거한다. 문자열 검사를 통해 텍스트를 제거하는데 제거하는 텍스트는 다음과 같다.

1. (숫자), ◎ 글머리
2. “잔류물의 정의”

이렇게 추출된 텍스트들을 리스트의 형태로 병합하고, 중복을 제거한다. 농약(동물용의약품)의 용도 같은 경우 문서마다 존재하는 문서가 있고 아닌 문서가 있기 때문에 리스트의 형태는 달라질 수 있다. 텍스트 리스트의 형태는 다음과 같다.

[농약명, (용도), 정의]

다음 단계는 테이블 추출 단계이다. 이 단계에서는 테이블 데이터를 추출하여 리스트로 병합하는 작업을 수행한다. 여기에서 추출하는 데이터는 그림 4 와 같다

식품명	mg/kg	식품명	mg/kg	식품명	mg/kg
소근육	0.1	돼지간	2.0	가금신장	0.1
소간	2.0	돼지지방	0.1	가금지방	0.1
소지방	0.1	돼지신장	5.0	유	0.2
소신장	5.0	가금근육	0.1	넙치	0.1
돼지근육	0.1	가금간	0.1	송어	0.1
잉어	0.1	양근육	0.1	말근육	0.1
염소근육	0.1	양간	2.0	말간	2.0
염소간	2.0	양지방	0.1	말지방	0.1
염소지방	0.1	양신장	5.0	말신장	5.0
염소신장	5.0				

그림 4. 테이블 데이터

테이블에서 추출된 데이터를 재정렬하여 dictionary 형태의 배열로 저장한다.

여기까지 추출된 텍스트, 테이블 데이터를 병합하는 작업이 필요하다. 인덱스 별 검사를 수행하여 테이블 데이터를 텍스트 리스트에 병합한다. 병합된 최종 리스트의 형태는 다음과 같다.

[농약명, (용도), 정의, {적용식품:잔류허용기준} list]

Ⅲ. 결론

본 논문에서는 잔류물질정보 웹 페이지의 개선을 위해 개정고시 문서 자동업데이트 시스템을 제안하였다. 제안된 시스템을 통해 수작업으로 진행하던 개정고시 업데이트를 자동으로 수행할 수 있어 작업의 효율이 증대 될 것으로 기대된다.

향후 연구로는 공백을 채우기 위한 데이터 수집, 데이터베이스 개선을 위한 효율적인 쿼리 작성을 진행할 예정이다.

참 고 문 헌

[1] “식품안전나라.” 잔류물질정보.
https://www.foodsafetykorea.go.kr/residue/main.do

Knowledge Plane to Auto-scale Next-generation Networks

Asif Mehmood*, Muhammad Saqib, Afaq Muhammad, Wang-Cheol SONG°
Jeju National University Jeju-si, Jeju-do, South Korea

asif@jejunu.ac.kr *, saqib@jejunu.ac.kr, afaq@jejunu.ac.kr, philo@jejunu.ac.kr °

Abstract

Machine learning (ML) techniques have been widely used for network controls and operations by the research communities. These techniques need to be improved at the Edge cloud for optimal decision making. As Edge cloud plays a very crucial role to fulfill peak demands enforced by the Fifth Generation (5G) use-cases. Therefore, a machine learning based knowledge plane has been shown, integrated with the Software-Defined Networking (SDN) and Network Analytics (NA) to perform efficient autoscaling for next generation networks.

Ⅰ. Introduction

Knowledge defined networking, comprised of a Knowledge Plane (KP) [1] plays an integral role in the field of networking which is a collection of multiple ML models designed by the Data Scientists after investing a considerable time of analysis.

Edge computing is very useful in different type of use cases as it comes with the benefits of latency reduction for the end users, while on the other hand it provides the service providers to distribute its load across the Edge nodes which enhances the service provisioning.

Edge computing frameworks still require to be developed and improved in the field of networking. Cloud Platforms such as Mobile Central Office Re-Architected Datacenter (M-CORD) [2] provide the ability to onboard the network services via Topology Orchestration Specification for Cloud Applications (TOSCA) configurations. While the management of the Edge frameworks can be done by the MEC Platforms and one such example is Aether; a management framework provided by Open Networking Foundation (ONF).

ML techniques has been very handy in the last years and made an effective change in the field of networking, and have been adopted widely to automate the operations related to lifecycle management of the services at the core, and as effective will be for the access networks [3].

Bringing up an idea that is based on a KP at the Edge, having a variety of ML techniques/models can evolve the future networks at the Edge. This would be useful as there is a diverse set of heterogenous devices connected to the multiple access technologies nearby the Edge locations. So, providing an intelligent layer of knowledge at the Edge could enhance the prediction decisions at the Edge as well will provide a lot more support for the 5G use cases.

This section puts a light on the Introduction and overview of the terminologies used. Afterwards, we detail our proposal in the Section II. We explain about the modules involved in our system. Then we conclude our remarks in Section III, in the form of benefits that could be integrated into an Edge cloud. We also explain our future intentions in this Sections. The remaining paper is as follows:

Ⅱ. Proposed Architecture

In this section, we propose an architecture shown in Fig. 1, based on a Knowledge Plane [1], an Edge Computing Infrastructure comprising of network functions, networks and devices, and a Network Analytics application. Integrating these three makes up a useful architecture that provides a platform to provide promising Edge usecase solutions.

Overall, we have the whole system comprising of three layers. First being the Knowledge Plane, contains the ML models trained by the data scientists. A data store is shown in the layer through which the scientists get and analyze the data from realtime datasets. After analyzing the data, a ML model is created, trained and tested until it becomes able to be used. After acceptance of model, it is then deployed onto the KP.

The KP is deployed across all of the edge nodes and is updated on the basis of all data collected from the edge nodes spanning all network edges. It then becomes of use for the network controllers to communicate on the northbound interface via Application Programmable Interfaces (APIs). This kind of Knowledge Plane is necessary to be used as there are a variety of rapidly growing, number of use cases at the Edge with the invention of flexible and adoptable future networks.

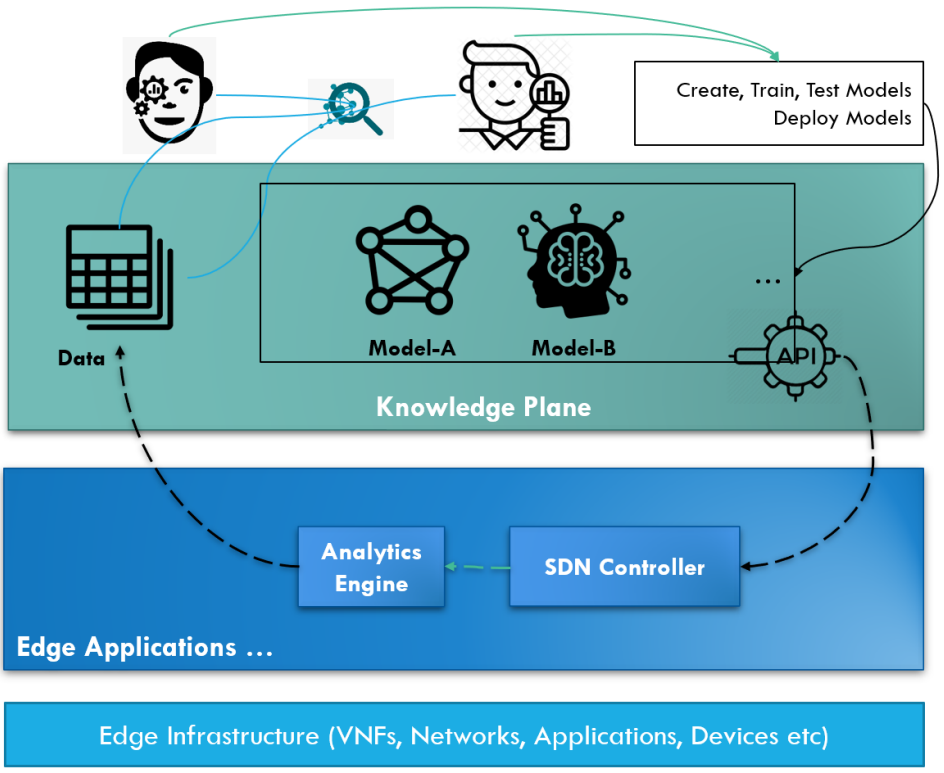


Figure 1: Proposed Architecture

It has also been shown in the Fig. 1 that the requests are made via REST APIs which makes the KP to decide which version of the available model to use at the time of request. Even if the KP takes time to be updated, the KP is able to utilize the current versions of the model running on the specified plane. It is just a matter of configurations, which could be reflected onto the KP after the model becomes available.

Secondly, we have an Edge Application layer where we have an SDN controller deployed integrated with a Network Analytics application which posts network analytics to the data store residing at the KP [1].

At the third layer, we have Infrastructure comprising of computer resources, network functions, networks, devices that make the use of these services/functions deployed at the Edge.

Use cases to be supported at the Edge require a faster and less latent response and is the reason why we designed the architecture in this way. ML models having the most critical role, enhance the auto-scaling process of the network functions at the Edge, the reason being that the Edge resources need to be allocated carefully. ML models on the basis of predictions, suggest the number of resources required for a network function serving a specific use case.

Ⅲ. Conclusion and Future

In this work, we proposed a state-of-the-art auto-scaling approach for next-generation networks at the Edge. For this, we proposed a ML-based KP. We are planning to use the network traffic statistics collected from the Edge locations, to predict the auto-scaling of resources. For example, the number of instances of a network service to be provisioned in the upcoming interval. On the basis of experiments to be made, we will analyze and compare the resource utilization for the Edge server.

ACKNOWLEDGMENT

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2017-0-01633) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A1B01016322).

References

[1] Clark, David D., et al. "A knowledge plane for the internet." Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications. 2003.

[2] Canellas Cruz, Ferran, Angelos Mimidis Kentis, and José Soler. “VIM Adaptation Layer for CORD.” (2018): n. pag. Print.

[3] Zhu, Guangxu, et al. "Toward an intelligent edge: wireless communication meets machine learning." IEEE Communications Magazine 58.1 (2020): 19-25.

네트워크 텔레메트리 기반 통합 네트워크 관리 시스템 연구

남석현°*, 임지윤*, 유재형*, 홍원기*

*포항공과대학교 컴퓨터공학과
{obiwan96, limjiyoon, jhyoo78, jwkhong}@postech.ac.kr

Network Telemetry-based Integrated Network Management System

Sukhyun Nam°*, Jiyoon Lim*, Jae-Hyoung Yoo*, James Won-Ki Hong*

*Department of Computer Science and Engineering, POSTECH

요 약

실시간으로 패킷 단위의 세부적인 네트워크 정보를 제공하는 INT (In-band Network Telemetry) 를 이용하면 우수한 성능으로 실시간 네트워크 이상 탐지와 로드 밸런싱이 가능하다. 본 논문은 INT 기반 통합 네트워크 관리 시스템을 제안하며, 제안한 시스템은 INT 를 이용하여 네트워크 상태 정보를 추출하며, 추출한 정보를 머신러닝을 이용하여 분석한다. 또한 실시간으로 네트워크 공격을 탐지하고 탐지된 공격에 대해선 완화 작용을 할 수 있으며, 네트워크 공격이 존재하지 않을 때는 네트워크 경로 별 가중치를 조정하여 네트워크 이용률을 높인다.

Ⅰ. 서 론

SDN (Software Defined Networking) 기술은 네트워크 관리에 있어 많은 것을 가능하게 한다. 네트워크 추상화, 네트워크 제어 기능의 논리적 중앙화가 SDN 의 중요한 특징이지만 특히, 최근에는 네트워크를 구성하는 스위치를 프로그래머블하게 만들어 컨트롤러에서 스위치에 대한 세부 조작이 가능하게 되었다. 프로그래머블 스위치의 언어로서 개발된 P4 [1]는 기존의 단순한 제어 평면의 트래픽 전달 동작 제어뿐만 아니라 각 스위치에서 헤더 포맷을 새로 정의하고 파싱하거나 match-action 테이블을 조절하는 등 더 복잡한 행동을 가능하도록 한다. P4 를 이용해 개발된 응용 기능 중 하나인 INT (In-band Network Telemetry)[2]는 실시간으로 네트워크의 상태 정보를 데이터 패킷에 실어 전달할 수 있어 네트워크에 대한 정보를 더 상세히 파악할 수 있도록 한다.

네트워크 관리에 있어서 가장 중요한 것은 로드 밸런싱과 네트워크 이상 탐지이다. 로드 밸런싱은 네트워크의 이용률을 높이기 위해 트래픽을 분산시켜 병목현상을 방지하는 기술이다. 네트워크 이상 탐지는 네트워크 상의 플로우에 대한 정보를 수집하여 네트워크에서 발생하는 악의적인 공격을 실시간으로 탐지하는 기술이다.

본 논문에서는 INT 를 이용한 네트워크 통합 관리 시스템을 제안한다. INT 를 활용하면 네트워크 상태에 대한 상세한 정보를 실시간으로 수집할 수 있어 로드 밸런싱과 이상 상태 탐지에 사용하기에 매우 적합하다. 따라서 INT 기반 네트워크 관리 시스템은 기존의 시스템보다 우수한 성능을 가진다.

본 논문의 구성은 다음과 같다. 2 장에서는 기존의 네트워크 이상 탐지 기술과 로드 밸런싱 기술, 그리고 INT 기술에 대해 기술한다. 3 장에서는 INT 를 활용한 네트워크 통합 관리 시스템 구조를 제안한다. 마지막으로 4 장에서는 결론을 기술한다.

Ⅱ. 관련 연구

1. 이상 상태 탐지 기술

NIDS (Network Intrusion Detection System)은 네트워크 상의 유해한 공격을 탐지하고 대응하는 시스템이다. NIDS 는 크게 두 가지 방식으로 연구되었는데, 첫째는 시그니처 기반(signature-based) 방식이고 둘째는 이상 상태 기반(anomaly-based) 방법이다. 두 방법은 현재의 네트워크 피처를 수집해서 기존에 학습된 이상 상태의 특징과 비교하느냐, 정상 상태의 특징과 비교하느냐 에 차이가 있다. 이상 상태의 특징과 비교하여 네트워크 공격을 탐지하는 시그니처 기반 방법은 학습 데이터에 없는 새로운 유형의 공격은 탐지하지 못하는 단점이 있어 최근에는 정상 상태와 비교하는 이상 상태 기반 탐지 기술이 주로 연구되고있다[3].

이상 상태 기반 탐지 기법은 주로 플로우 기반(flow-based)으로 연구가 이루어지고 있다. 현재 플로우의 특징을 수집하여 기존의 정상 상태 데이터 셋과 얼마나 다른가를 판단하기 위해 초기에는 Fuzzy logic 기반의 연구가 존재하였다[4]. 해당 연구는 DoS 공격에는 94%의 높은 F1 score 를 보였으나 그 외의 공격에는 낮은 탐지율을 보였다. 최근에는 분류(Classification) 알고리즘을 적용한 연구들이 주를 이루었다. SVM (Support Vector Machine)을 이용하여 정확도 98.29%를 기록한 연구가 존재한다[3]. 인공 신경망을 이용한 연구도 활발히 진행되고 있는데, NSL-KDD 데이터 셋의 네 가지 공격을 탐지 대상으로 심층 신경망(Deep Neural Network)을 사용하여 F1 score 75.57%의 성능을 보여준 연구가 존재한다[5].

해당 연구들은 주로 인입 포트 및 TCP 플래그 정보를 사용하였기 때문에 현재의 네트워크의 상태에 대한 자세한 정보를 사용하지 못하였다. NSL-KDD 데이터 셋을 사용한 연구의 경우에는 다양한 피처가 존재하여

비교적 높은 성능을 보이긴 하나 이는 네트워크 상에서 실시간으로 수집할 수 없는 정보이다. INT 를 이용하면 네트워크의 정보를 실시간으로 상세히 수집 가능하며, 이를 이용한 선행연구에서는 DoS 공격 탐지에 대해 92.41%의 높은 F1 score 를 보였다[6].

2. 로드 밸런싱 기술

데이터 센터에서 주로 사용하고 있는 로드 밸런싱 알고리즘으로는 ECMP (Equal-cost Multipath)[7]가 있다. ECMP 는 출발지에서 목적지까지의 최단 경로가 여러 개일 때 해당 경로들에 플로우를 균등하게 분배하는 알고리즘이다. 하지만 이는 데이터 센터에 여러 개의 대용량 트래픽이 생성될 경우 같은 경로로 할당되는 문제가 있어 네트워크의 혼잡 정도를 실시간으로 파악하고 이를 로드 밸런싱에 활용하기 위한 연구들이 지속되고 있으며, 해당 연구들은 일반적으로 성능 측정을 할 때 FCT (Flow Completion Time)을 ECMP 와 비교한다.

각 스위치에 이웃한 노드의 트래픽 혼잡 정도를 저장하여 혼잡 정도가 가장 적은 다음 경로를 지정하여 평균적으로 ECMP 에 비하여 1.2 배의 성능을 보인 연구가 있다[8]. 프로그래머블 스위치를 활용하여 주기적으로 탐지 패킷(probe packet)을 전송하여 각 스위치 별로 최적의 다음 홉 정보를 저장하여 평균적으로 ECMP 대비 1.52 배의 성능을 보인 연구도 있다[9].

이러한 연구들은 모든 스위치에 네트워크 전체의 트래픽 혼잡 정보를 저장하여 일반적인 스위치에서는 사용할 수 없는 문제나 추가 패킷을 생성하여 네트워크에 오버헤드를 일으키는 단점이 있다. In-band 네트워크 텔레메트리 기법을 활용하면 추가적인 패킷을 생성하지 않고 P4 스위치를 사용하여 네트워크 상태 정보를 파악할 수 있다[10].

3. INT (In-band Network Telemetry) [2]

INT 는 P4 를 활용한 네트워크 모니터링 프레임워크로, 패킷 헤더에 INT 헤더를 정의하여 별도의 탐지 패킷을 생성하지 않고 네트워크 정보를 수집할 수 있다. P4 프로그램을 통해 데이터 평면 상에 INT 헤더를 삽입하는 Source 스위치, INT 헤더에 네트워크 정보를 삽입하는 Transit 스위치, INT 헤더를 추출하여 전달하는 Sink 스위치를 정할 수 있다. INT 를 통해 수집 가능한 정보는 스위치 ID, 인입 및 인출 포트 관련 정보, 플로우 지연 정보, 스위치 큐 사용량 등이 있다.

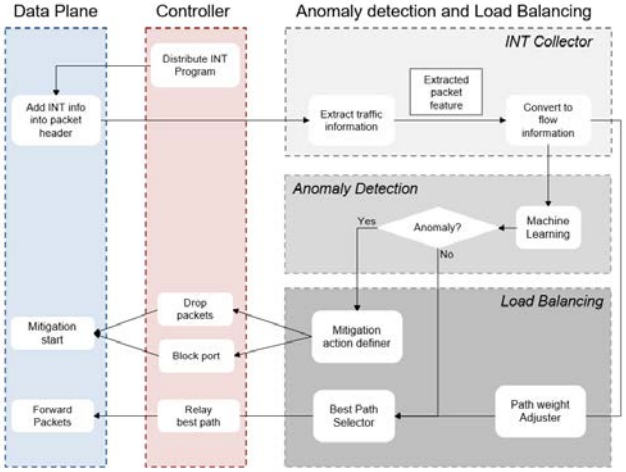


그림 1 INT 기반 네트워크 통합 관리 시스템

Ⅲ. INT 기반 통합 네트워크 관리 시스템

그림 1 은 본 연구에서 제안하는 INT 기반 네트워크 통합 관리 시스템을 나타낸다. 해당 시스템은 INT 를 기반으로 만들어져 실시간으로 네트워크에 대한 관리를 진행할 수 있다. 각 모듈에 대한 상세한 설명은 다음과 같다.

1. INT Collector

INT Collector 는 데이터 평면에서 수집된 패킷 별 INT 정보를 플로우 정보로 변환한다. INT 정보가 수집되는 알고리즘은 다음과 같다. 컨트롤러에서 수집 대상 INT 정보를 정의하여 데이터 평면에 배포한다. 데이터 평면상의 스위치들은 수집 대상 INT 정보를 패킷 헤더에 포함시켜 패킷을 전송한다. Sink 스위치에서 INT 패킷 헤더와 수집 대상 INT 정보를 추출하여 INT 정보만 INT Collector 로 전달한다. 하지만 전달된 데이터는 패킷 별 데이터이기 때문에 플로우 별 데이터로 전환해야 한다. 플로우는 같은 출발지와 목적지를 가지는 일정 시간 이내의 모든 패킷으로 정의된다. INT Collector 는 정의에 따라 패킷 데이터를 플로우 별로 모으고 각 피처를 플로우 별로 평균값을 계산하여 플로우 별 데이터로 전환한 후 Anomaly Detection 모듈과 Load Balancing 모듈로 전달한다.

2. Anomaly Detection

Anomaly Detection 모듈에서는 수집된 피처를 이용하여 해당 플로우가 이상 상태인지 아닌지를 판별한다. Anomaly Detection 모듈에는 사전에 학습된 기계 학습 모듈이 내장된다. 기계 학습 모듈은 INT Collector 에서 전달받은 플로우 별 데이터 입력 피처로 하여 이상 상태 점수를 반환하도록 학습된다. 이 때 사용하는 기계 학습 모델은 가장 많이 쓰이는 지도 학습(Supervised Learning) 모델 중 하나인 순환 신경망 (Recurrent Neural Networks)을 사용한다. 이상 상태 점수는 0 에서 1 사이의 값이다. 이상 상태일 경우 이상 상태 점수가 Load Balancing 모듈로 전달되어 이용될 수 있도록 한다.

Anomaly Detection 모듈에 대한 선행 연구에서는 네트워크 플로우 별로 INT 정보만 수집하여 순환 신경망에서 학습하였다. INT 정보와 함께 TCP 플래그와 출발 및 도착 포트 정보를 함께 사용하면 Anomaly Detection 모듈에서 네트워크 공격에 대해 F1 score 를 95% 이상의 성능을 낼 수 있을 것으로 보인다.

3. Load Balancing

Load Balancing 모듈은 이상 플로우에 대한 완화 작용 선택 모듈 (Mitigation action definer)을 포함하고 있다. 완화 작용 선택 모듈은 Anomaly Detection 모듈에서 이상 상태임을 전달받으면 해당 플로우에 대한 로드 밸런싱을 중지하고 이상 상태 점수에 따른 완화 작용을 선택할 수 있도록 한다. 완화 작용 선택에는 이상 상태 점수를 이용하는데, 이상 상태 점수가 높을 경우 해당 출발 포트를 차단하고, 낮을 경우에는 해당 플로우만 패킷 드랍(packet drop)하는 방식을 선택하여 이상 상태 정도에 맞는 완화 작용을 선택하여 컨트롤러로 전달한다. 컨트롤러는 Load Balancing 모듈에게 전달받은 완화 작용을 해당 플로우에 대해 실행한다.

Load Balancing 모듈은 이상 상태로 판별되지 않은 플로우들에 대해서 경로 가중치 계산을 한다. 가중치 계산에는 INT Collector 에서 전달받은 플로우 별 데이터를 활용한다. Load Balancing 모듈은 사전에

학습된 기계 학습 모델을 이용한다. 해당 기계 학습 모델은 강화 학습(Reinforcement Learning) 기법을 사용한다. 강화 학습 모델은 INT 정보 중 네트워크 각 링크의 처리율(throughput)을 입력 피처로 사용하여 각 출력 포트에 대한 가중치를 반환하도록 학습된다. 이 때 출력 포트에 대한 가중치들은 스위치 별로 가중치의 합이 1 이 되도록 한다. 강화 학습 모델을 학습시킬 때 FCT 를 최소화 시키도록 학습시킨다.

Load Balancing 모듈은 강화 학습의 결과를 컨트롤러에 전달한다. 컨트롤러는 출력 포트들에 대한 가중치를 스위치들에 전달하여 스위치들에서 해당 가중치에 해당하는 비율로 경로를 설정하도록 한다. 강화 학습에서 입력 피처로 네트워크 전체 정보를 사용하였기 때문에 네트워크 일부의 환경이 변하여도 전체 네트워크가 빠르게 대응할 수 있도록 한다. 또한 강화 학습을 통해 주어진 네트워크 환경에서 데이터 수집과 학습을 계속하게 되기 때문에 ECMP 대비 1.2 배 이상의 성능을 보일 수 있을 것으로 보인다.

IV. 결론

본 연구에서는 실시간으로 패킷 단위의 세부적인 네트워크 정보를 수집할 수 있는 INT 를 이용하여 네트워크 이상 상태 탐지와 로드 밸런싱을 모두 할 수 있는 INT 기반 통합 네트워크 관리 시스템을 제안하였다. 해당 시스템은 이상 상태 탐지와 로드 밸런싱에 각각 기계 학습을 적용하여 높은 성능이 기대되며, 특히 두 기능을 모두 포함하고 있기 때문에 네트워크 공격이 일어나면 로드 밸런싱에 이를 활용하여 완화작용까지 진행할 수 있는 이점을 가지고 있다.

ACKNOWLEDGMENT

이 논문은 2020 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발)

참 고 문 헌

[1] P. Bosshart *et al.,* "P4: Programming Protocol-Independent Packet Processors." ACM SIGCOMM Coumputer Communication Review, vol. 44, no.3, pp. 87-95, 2014.

[2] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, "In-band Network Telemetry via Programmable Dataplanes," ACM SOSR, 2015, pp. 2-3.

[3] P. Winter, E. Hermann, and Z. Markus, "Inductive Intrusion Detection in Flow-Based Network Data using One-Class Support Vector Machines," IFIP International Conference on New Technologies, 2011, pp. 1-5.

[4] R. Shanmugavadivu, N. Nagarajan, "Network intrusion detection system using fuzzy logic," Indian Journal of Computer Science and Engineering, 2011, pp.101- 111.

[5] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi,and M. Ghogho, "Deeplearning approach for Network Intrusion Detection in Software Defined Networking," WINCOM, 2016, pp. 258-263.

[6] 임지윤, 남석현, 유재형, 홍원기, "INT 기반 네트워크 이상 상태 탐지 기술 연구", KNOM Review, Vol. 22, No. 3, December 2019.

[7] M. Chiesa, G. Kindler, and M. Schapira, "Traffic Engineering with Equal-Cost-MultiPath: An Algorithmic Perspective," IEEE Conference on Computer Communications, 2014.

[8] M. Alizadeh et al., "CONGA: distributed congestion-aware load balancing for datacenters," Proceedings of the 2014 ACM conference on SIGCOMM,2014.

[9] N. Katta, M. Hira, C. Kim, A. Sivaraman, and J. Rexford, "HULA: Scalable Load Balancing Using Programmable Data Planes," Proceedings of the Symposium on SDN Research,2016.

[10] 임지윤, 현종환, 유재형, 홍원기, "머신러닝 기반 동적 경로 가중치 조정 로드밸런싱 알고리즘 연구," KNOM Conference 2019, Daegu, Korea, May. 30, 2019, pp. 83-86.

엣지 컴퓨팅 환경에서 기계 학습 기반의 효율적인 침입 탐지 시스템 연구

이종화, 방지원, 최미정*
강원대학교
{a8478836, jiwonbang, mjchoi}@kangwon.ac.kr

Efficient Intrusion Detection System based on Machine Learning in Edge Computing Environment

Jong-Hwa Lee, Jiwon Bang, Mi-Jung Choi*
Dept. of Computer Science, Kangwon National Univ.

요 약

엣지 컴퓨팅은 클라우드 컴퓨팅 환경에서 높은 지연 시간, 낮은 확장성, 대용량의 컴퓨팅 리소스 요구 등과 같은 문제점을 해결하기 위해 등장한 새로운 네트워크 모델이다. 그러나 엣지 컴퓨팅은 클라우드 컴퓨팅 환경이 가지고 있던 고질적인 보안 문제는 해결하지 못한다. 오히려 새로운 프로토콜들이 추가되고 다양한 기기들이 연결되면서 복잡해진 컴퓨팅 환경으로 인해 새로운 보안문제가 발생하였고, 이로 인해 기존의 침입 탐지 시스템으로는 복잡하게 분산된 환경에서의 침입 탐지가 어려운 일이 되었다. 최근 연구들에서는 이런 문제들을 해결하기 위하여 기계 학습 기법을 침입 탐지 시스템에 도입하였으며, 기존의 침입 탐지 시스템보다 높고 정확한 탐지율을 보여주고 있다. 본 논문은 엣지 컴퓨팅 환경에서 더 빠르고 효율적인 기계 학습 기반의 침입 탐지 시스템을 설계하고자 한다.

Ⅰ. 서 론

4차 산업혁명으로 인해 다양하고 많은 수의 IoT 기기들이 생겨나면서 네트워크에 연결하는 장치들의 수와 그에 따른 데이터의 양과 트래픽 또한 급격히 증가하고 있다. 기존의 중앙 집중 방식의 클라우드 서버는 수십억 개에 이르는 단말 장치들이 전송하는 데이터들을 처리하기 위해 대용량의 컴퓨팅 리소스를 요구하며, 이에 따라 네트워크상에서 데이터의 처리 응답 시간이 길어지게 되었고 확장성과 이동성 등을 만족할 수 없게 되었다. 이러한 문제들을 해결하기 위하여 기존의 클라우드 컴퓨팅에서 클라우드 계층과 클라이언트 계층 사이에 엣지 계층을 추가하여 데이터들을 클라우드 서버로 보내지 않고 단말 장치들과 가까운 엣지 계층에서 사용자의 요청들을 실시간으로 처리하는 엣지 컴퓨팅이 등장하였다[1, 2].

Graphics Processing Unit(GPU) 장치, ReLU와 같은 알고리즘 그리고 학습할 수 있는 데이터의 증가로 인해 다양한 분야에서 Machine Learning(ML)을 도입하여 연구가 진행되고 있다[3]. 기존의 침입 탐지 시스템(Intrusion Detection System: IDS)은 IoT와 같은 복잡한 환경에서 기하급수적으로 늘어나는 데이터와 끊임없이 변이하는 악성코드들을 처리하기에는 역부족이다[4]. 이에 따라 기계 학습을 사용하여 기존의 IDS의 단점들을 극복하고 높은 정확도와 효율성으로 다양한 사이버 공격을 탐지하는 연구들이 진행되고 있다[5].

현대 네트워크와 시스템을 위협하는 새로운 보안 위협 및 변종들은 다양한 프로토콜과 신기술의 추가 등으로 인해 이전의 공격들이 변형되어 생겨났으며, 기존의 보안 시스템은 이러한 공격들을 탐지하는 데에 어려움을 맞이하고 있다[6, 7]. 엣지 컴퓨팅 환경의 특성으로 인해 다양한 서로 다른 기기들이 액세스 포인트에 연결됨에 따라 몇몇 기업들은 데이터 센터를 여러 군데에 설치하였다. 설치과정에서 다양한 기술이 적용되면서 취약점들이 등장하였고 그로 인해 보안 문제들이 발생했다[8, 9]. 따라서 본 논문에서는 엣지 컴퓨팅 환경에서 기계 학습으로 효율적인 사이버 공격을 탐지하는 모델을 설계하고자 한다. 2장에서는 엣지 컴퓨팅, 기계 학습, IDS와 관련된 연구들에 대해 설명하고 3장에서는 본 논문에서 제안하는 모델을 설계하고 4장에서는 결론을 내리면서 논문을 마친다.

Ⅱ. 관련 연구

1) 엣지 컴퓨팅

엣지 컴퓨팅은 유연한 컴퓨팅 기술을 제공해주지만 다양한 보안 취약점들을 가지고 있다. 예시로는 클라우드 서버에 비해 약한 컴퓨팅 능력, UI의 부재로 인한 공격의 미인지, OS와 프로토콜의 이기종성 등이 있다. 제시한 취약점들을 이용한 공격들의 종류에는 Denial of Service(DoS), Man In The Middle(MITM), Spoofing, Advance Persistent Threats(APT), Account Hijacking 등이 있다. 이 뿐만 아니라 단말 기기와 엣지 서버, 엣지 서버와 클라우드 서버간의 통신 보안의 취약점도 존재한다[10, 11]. 따라서 사용자 인증이 필요하고, 믿을 수 있는 통신 모델을 사용하여 사용자와 단말 기기간의 데이터들은 적절한 암호화가 진행되어야 한다. 인증 측면에서 얼굴 인식이나 지문 인식과 같은 생체 인증 기술은 기존의 Public Key Interface(PKI) 기반의 키 교환 방식의 비효율적이고 낮은 확장성 등의 문제점을 해결할 수 있다. 또한, 통신 데이터들은 모두 무결성을 검증할 수 있어야하고 누구나 볼 수 없도록 해야 하며 믿을 수 없는 서버나 사용자로부터의 통신 요청은 확인 후 수락이 되어야 한다[12].

2) 기계 학습

기계 학습은 많은 양의 주어진 데이터에서 패턴을 인식하거나 연속적인 어떤 값을 추정해야하는 회귀 문제를 컴퓨터를 이용하여 해결하는 기술이

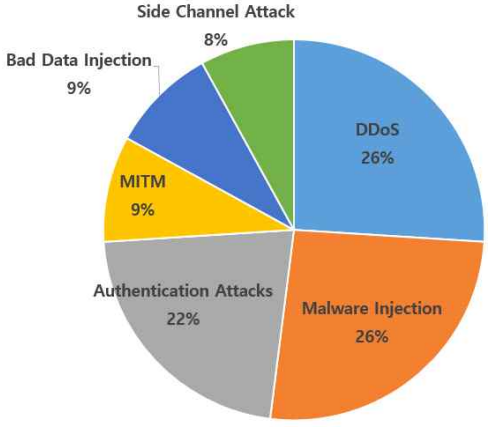
다. 기계 학습 방법에는 지도 학습, 비지도 학습, 강화 학습이 있다. 지도 학습은 입력과 출력에 대한 정보를 모두 알고 있는 상태에서 학습이 진행되며 입력들을 분류하거나 특정 값을 예측하는 데에 사용된다. 지도 학습은 그 특성으로 인해 모델의 불필요한 데이터로 인해 특성이 과도하게 반영된 과적합(over-fitting) 문제와 모델이 간단하여 필요한 데이터의 특성이 너무 적게 반영된 부적합(under-fitting) 문제가 발생할 수 있다. 따라서 이러한 문제들이 발생하지 않도록 모델을 설계하는 것이 중요하다. 비지도 학습은 지도 학습과 다르게 입력과 출력에 대한 정보가 필요하지 않다. 따라서 비지도 학습은 정답이 없는 입력 값으로 모델이 학습 데이터에서 숨겨진 특성을 발견하여 스스로 학습하고 분류하는 방법이다. 강화 학습은 앞선 두 개의 학습과 다르게 환경에 대한 데이터를 입력으로 받아 시행착오를 반복하면서 데이터에 대한 출력이 올바른지에 대한 여부를 확인하고 다시 수정하는 학습 방법이다[13].

3) IDS

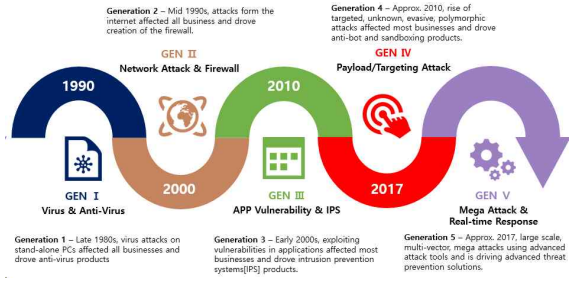
IDS의 목표는 시스템이나 네트워크에서 의심스러운 행위들을 미리 정의된 규칙에 따라 발견하고 관리자 또는 사용자에게 알려주는 것이다. IDS의 종류에는 크게 두 가지가 존재한다. 네트워크 기반의 Network based IDS(NIDS)와 호스트 기반의 Host based IDS(HIDS)가 있다. NIDS는 각각의 패킷들을 분석하는 반면에 HIDS는 호스트의 모든 행위들을 감시한다[14]. IoT 환경 특성상 효과적으로 보안 위협들을 탐지하기 위해서 IDS는 좀 더 간단하고, 실시간으로 정확하게 탐지가 되어야 한다. 세계적으로 널리 사용되는 IDS지만 오탐이 많다는 단점을 가지고 있다. 오탐 문제를 해결하기 위해 기계학습을 이용하여 오탐을 줄이고 정확도를 높이는 연구가 진행되고 있다[15, 16].

Ⅲ. 시스템 설계

엣지 컴퓨팅 환경에서 직접적으로 발생하는 대다수의 공격은 (그림 1)과 같다. Distributed Denial of Service(DDoS)와 Malware Injection 공격이 26%로 대부분을 차지하며 Authentication Attacks가 22%, MITM과 Bad Data Injection이 9%, Side Channel Attack이 8%를 차지했다. 또한, (그림 2) 처럼 공격들은 점점 다양화되고 지능화 되고 있다. 초기 악성 코드들은 네트워크가 발전하면서 감염력이 증가하기 시작했고 컴퓨터 뿐 만이 아닌 모바일 장치나, IoT 장치들까지 감염시키면서 많은 피해를 주고 있다. 하지만 이런 악성 코드들을 완벽하게 잡아낼 수 있는 프로그램이나 시스템은 턱 없이 부족하다[17]. 따라서 엣지 컴퓨팅 환경에서 높은 정확도로 빠르고 효율적인 침입 탐지가 가능한 시스템 연구가 필요하다.



(그림 1) 엣지 컴퓨팅 환경의 공격 유형.

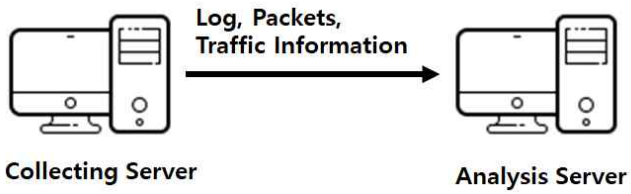


(그림 2) 사이버 보안 기술의 진화과정 및 특성.

기계 학습 모델 중 지도 학습을 위해 공공 데이터세트인 NSL-KDD 데이터세트와 UNSW-NB15 데이터세트를 사용한다. 두 개의 공공 데이터세트는 신뢰할 수 있는 기관에서 실제 데이터들을 가공하여 만들었으며 다양한 연구에서 높은 성과를 낸 대표적인 데이터세트이다. 또한, 공공 데이터세트를 사용함으로써 충분한 학습 데이터가 없을 경우 양질의 데이터를 사용할 수 있는 이점이 있다. 네트워크 침입 탐지를 위해 사용했던 KDDCUP99 데이터세트에는 학습 데이터와 시험 데이터에 각각 78%, 75%의 중복 된 데이터로 인해 학습 결과를 편향적으로 만드는 문제점이 있었다. NSL-KDD 데이터세트는 KDDCUP99 데이터세트의 중복된 레코드들을 삭제하여 개선한 데이터세트이다. NSL-KDD 데이터세트는 43개의 특징이 있으며 공격 유형을 DoS, Probe, Remote to Local(R2L), User to Root(U2R)로 분류하였다[18]. UNSW-NB15 데이터세트는 KDDCUP99와 NSL-KDD 데이터세트가 현대의 네트워크의 특성을 반영하지 못하기 때문에 만들어진 데이터세트이다. UNSW-NB15 데이터세트에는 49개의 특징이 있으며 공격 유형을 Fuzzers, Backdoors, DoS, Exploits 등의 9개로 분류하였다[19].

지도 학습과 다르게 비지도 학습은 정답 데이터가 없어도 학습이 가능하다. 따라서 비지도 학습을 위해서 Autoencoder 모델을 사용하여 학습을 진행한다. Autoencoder 모델은 내부 네트워크를 목적에 맞게 변형함으로써 다양한 응용이 가능하므로 응용 모델을 통해 최적의 탐지 모델을 찾아낼 수 있다. 본 연구에서 프로토타입에 사용할 Autoencoder 모델은 3계층으로 이루어진 Neural Network로 입력 층, 은닉 층, 출력 층으로 구성된다. Autoencoder는 고차원 데이터에서 저차원 데이터로 벡터를 변환시킨다. 입력 층에서 은닉 층으로 보내는 과정을 인코딩(encoding)이라하고, 은닉 층에서 출력 층으로 보내는 과정을 디코딩(decoding)이라고 한다. 인코딩과 디코딩을 거쳐서 출력된 값은 입력 값과 동일한 차원을 가진다. 데이터를 넣어주면 인코딩과 디코딩 과정을 통해 모델이 특징을 추출하여 스스로 학습을 하고 출력 결과의 오류를 수정하는 과정을 반복하면서 가장 최적화된 학습 결과가 나온다[20].

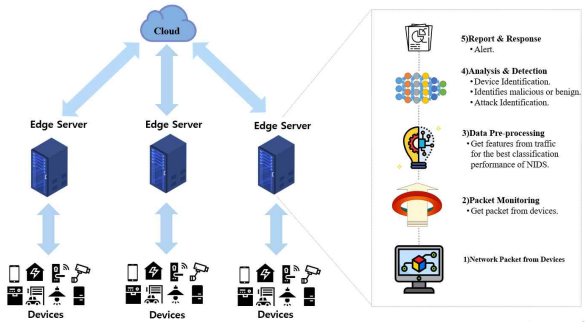
지도 학습의 시험 데이터와 비지도 학습의 입력 데이터 및 시험 데이터를 얻기 위해서 허니팟(honeypot)을 (그림 3)과 같이 구성한다. 허니팟은 네트워크에 존재하는 악성코드의 정보를 수집하기 위해 취약한 보안 설정을 사용한다[21]. 허니팟은 엣지 컴퓨팅 환경의 가장 최하단에 위치하며, 데이터들을 수집하고 분석 서버로 전송한다. 전송 받은 데이터들은 tcpdump, Nagios, pcap, Wireshark 등의 오픈 패킷 분석 툴들을 적극 활용하고, 기계 학습에 사용될 수 있도록 전처리한다.



(그림 3) 데이터 수집과 분석을 위한 허니팟 시스템.

기계 학습에서 아키텍처 최적화의 목표는 일반화의 오류(generalization error)를 최소화하는 것이다. 데이터세트를 학습 데이터, 시험 데이터로만 나누어 사용하면 일반화의 오류가 일어나 편향된 결과가 도출될 가능성이 높다. 따라서 일반화의 오류가 일어났는지 검증하여 모델을 수정할 수 있도록 학습 데이터와 시험 데이터뿐만 아니라 검증 데이터도 필요하다. 따라서 지도 학습은 공공 데이터세트에서 검증 데이터와 학습 데이터를 1:9, 2:8, 3:7로 나누어 학습을 진행하고 수집한 실제 데이터세트로 시험을 진행한다[22].

앞에서 제시한 사항들과 엣지 컴퓨팅 환경이 가지고 있는 보안 문제를 해결하는 효율적인 사이버 공격 탐지를 위한 기계 학습 기반의 시스템 모델을 소개한다. 본 시스템의 구성은 (그림 4)처럼 크게 세 부분으로 이루어져 있다. 가장 하위 층부터 차례대로 Device Layer, Edge Layer, Cloud Layer로 구성되어 있다. Device Layer에서는 다양한 기종의 IoT 장비들이 복잡하게 분포되어 있다. 각 장비들은 일정한 주기를 가지고 네트워크 트래픽이나 패킷과 같은 여러 정보들을 수집하여 Edge Layer의 Edge Server로 전송한다. 엣지 컴퓨팅 환경 구성을 위해서 오픈소스 기반의 엣지 컴퓨팅 플랫폼인 EdgeX Foundary, Azure IoT Edge 등을 활용할 수 있다. 데이터를 전송 받은 Edge Layer에서는 (그림 4)와 같은 5가지의 작업을 진행한다. 장치들로부터 네트워크 정보들을 수집, 수집한 정보들의 모니터링, 기계 학습에서 특징 추출을 위한 데이터 전처리, 전달 받은 최적화된 모델을 이용한 분석과 탐지, 발견한 악성 코드에 대한 보고와 대응을 한다. Cloud Layer에서는 Edge Layer에서 전달 받은 전처리된 데이터들을 사용하여 최적의 기계 학습 모델을 구축한다. 지도 학습을 사용할 경우 데이터세트는 널리 알려진 공공 데이터세트인 NSL-KDD 데이터세트와 UNSW-NB15 데이터세트를 사용하여 기계 학습 모델을 학습 시키며, 비지도 학습을 사용할 경우는 전처리된 데이터들로 모델을 학습한다. 모델 내부의 유닛들의 배치와 하이퍼파라미터들을 조정하면서 모델을 평가한 후 최적의 결과가 도출되면 모델의 상태를 저장하고 이를 다시 Edge Layer로 전송한다. Edge Layer에서는 전달 받은 학습이 완료된 기계 학습 모델을 이용하여 실제 데이터들을 정상과 공격 유형에 따라 분류한다. 만약, 의심스러운 공격이 탐지된다면, 공격 유형에 맞게 적절한 조치를 취하고 사용자에게 알리게 된다.



(그림 4) 최종 플랫폼 구성도.

Ⅳ. 결론

엣지 컴퓨팅은 무수히 늘어나는 IoT 장치들과 사용자에게 서비스를 제공하기 위한 클라우드 컴퓨팅의 지리적 문제점과 높은 통신 지연율 등의 문제를 해결하여 실시간 처리로 데이터 지속성에서 많은 이점을 가져다주었다. 하지만 굉장한 규모의 기종이 다른 다양한 장치들이 연결되면서 보안 공격의 대상이 되었다. 기존의 침입 탐지 시스템들은 침입 탐지에 있어서 오탐률이 높았으며 엣지 컴퓨팅 환경과 같은 단말 장치들이 분산된 환경에서 효율이 현저히 낮았다. 다양한 분야에서 두각을 나타낸 기계 학습을 이용한 IDS가 연구되었고, 엣지 컴퓨팅 환경에 적용하면서 높은 정확도와 고속의 침입 탐지가 가능하게 되었다. 본 논문에서는 엣지 컴퓨팅 환경에서 다양한 기계 학습 모델을 이용하여 효율적인 침입 탐지 시스템을 제안하였다.

향후 연구로 본 논문에서 설계한 모델을 구현하고 직접 공격을 수행하여 공격마다 최적의 기계 학습 모델을 구성하는지, 적절한 대응이 가능한지 검증한다. 또한, 지도 학습에서 NSL-KDD 데이터세트와 UNSW-NB15 데이터세트 이외에 계속해서 등장하는 다양한 공공 데이터세트를 사용하고 비지도 학습에서 Autoencoder 모델 이외의 학습 모델을 다양하게 사용한 비교 연구도 진행할 예정이다. 본 논문에서 제시한 오픈소스를 활용한 시스템 모델을 통해 초기 설계의 시간적 비용을 절감하고 엣지 컴퓨팅 환경에서 높은 정확도를 가진 침입 탐지 모델을 구성하여 엣지 컴퓨팅 환경의 보안 향상에 기여할 수 있을 것이라 기대한다.

ACKNOWLEDGMENT

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임.(NRF-2020R1A2C1012117).

참 고 문 헌

[1] HyunMoon Park and TaeHo Hwang, “Changes and trends in edge computing technology,” *Journal of The korean Institute of Communication Sciences*, Vol. 36, No. 2, pp. 41-47, Jan. 2019.

[2] Shanhe Yi, Cheng Li and Qun Li, “A Survey of Fog Computing: Concepts, Applications and Issues,” In *proc. of the 2015 Workshop on Mobile Big Data*, pp. 37-42, Jun. 2015.

[3] GwiHoon Kim and YongGeun Hong, “Machine Learning technology trends in the network,” *Journal of The korean Institute of Communication Sciences*, Vol. 34, No. 10, pp. 38-44, Sep. 2017.

[4] Chao Liang, Bharanidharan Shanmugam, Sami Azam, Mirjam

Jonkman, Friso De Boer and Ganthan Narayansamy, “Intrusion Detection System for Internet of Things based on a Machine Learning approach,” In *proc. of International Conference on vision Towards Emerging Trends in communication and Networking*, pp. 1-6, Mar. 2019.

[5] Nadai Chaabouni, Mohamed Mosbah, Akka Zemmari, cyrille Sauvignac and Parvez Faruki, “Network Intrusion Detection for IoT security Based on Learning Techniques,” *Journal of IEEE Communications Surveys & Tutorials*, Vol. 21, No. 3, pp. 2671-2701, Jul. 2019.

[6] Abebe Abeshu Diro and Naveen Chilamkurti, “Distributed attack detection scheme using deep learning approach for Internet of Things,” *Journal of Future Generation Computer Systems*, Vol. 82, pp. 761-768, May. 2018.

[7] Abebe Abeshu Diro and Naveen Chilamkurti, “Deep Learning: The Frontier for Distributed Attack Detection in Fog-to-Things Computing,” *Journal of IEEE Communications Magazine*, Vol. 56, No. 2, pp. 169-175, Feb. 2018.

[8] Mithun Mukherjee, Rakesh Matam, Lei Shu, Leandros Maglaras, Mohamed Amine Ferrag, Nikumani Choudhury and Vikas Kumar, “Security and Privacy in Fog Computing: Challenges,” *Journal of IEEE Access*, Vol. 5, pp. 19293-19304, Sep. 2017.

[9] Farhoud Hosseinpour, Payam Vahdani Amoli, Juha Plosila, Timo Hamalamen and Hannu Tenhunen, “An Intrusion Detection System for Fog Computing and IoT based Logistic Systems using a Smart Data Approach,” *Journal of Digital Content Technology and its Applications*, Vol. 10, No. 5, pp. 34-46, Oct. 2016

[10] Yinhao Xiao, Yizhen Jia, Chunchi Liu, Xiuzhen Cheng, Jiguo Yu and Weifeng Lv, “Edge Computing Security: State of the Art and Challenges,” *Journal of IEEE*, Vol. 107, No. 8, pp. 1608-1631, Aug. 2019.

[11] Saad Khan, Simon Parkinson and Yongrui Qin, “Fog computing security: a review of current applications and security solutions,” *Journal of Cloud Computing: Advances, Systems and Applications*, Vol. 6, No. 1, pp. 1-22, Aug. 2017.

[12] Shanhe Yi, Zhengrui Qin and Qun Li, “Security and Privacy Issues of Fog Computing: A Survey,” In *proc. of International Conference on Wireless Algorithms, Systems, and Applications*, pp. 685-695, Aug. 2015.

[13] Seong-Eun Moon, Soobeom Jang, JeongHyuk Lee and Jong-Seok Lee, “Trends in machine learning and deep learning technology,” *Journal of Korea Information and communications Society*, Vol. 33, No. 10, pp. 49-56, Nov. 2016.

[14] Ram Kumar Singh and T. Ramajujam, “Intrusion detection System Using Advanced Honeypots,” *Journal of Computer Science and Information Security*, Vol. 2, No. 1, pp. 1-9, Jun. 2009.

[15] Chen Jun and Chen Chi, ,“Design of Complex Event-Processing IDS in Internet of Things,” In *proc. of 2014 sixth international Conference on Measuring Technology and Mechatronics Automation*, pp. 10-11, Jan. 2014.

[16] Iksoo Shin, Jungsuk Song, Jangwon Choi and Taewoong Kwon, “A Practical Feature Extraction for Improving Accuracy and Speed of IDS Alerts Classification Models Based on Machine Learning,” *Journal of Korea Institute of Information Security and Cryptology*, Vol. 28, No. 2, pp. 385-395, Mar. 2018.

[17] Check Point, “Check Point Infinity Whitepaper,” https://www.checkpoint.com/architecture/infinity/, Retrieved Mar. 2020.

[18] S. Revathi and A. Malathi, “A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection,” *Journal of Engineering Research & Technology*, Vol. 2, No. 12, pp. 1848-1853, Dec. 2013.

[19] Nour Moustafa and Jill Slay, “UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems,” In *proc. of 2015 military communications and information systems conference*, pp. 1-6, Nov. 2015.

[20] Binghao Yan and Guodong Han, “Effective Feature Extraction via Stacked Sparse Autoencoder to Improve Intrusion Detection System,” *Journal of IEEE Access*, Vol. 6, pp. 41238-41248, Jul. 2018.

[21] Abhishek Mairh, Debabrat barik, Kanchan Verma and Debasish Jena, “Honeypot in Network Security: A survey,” In *proc. of 2011 international conference on communication, computing & security*, pp. 600-605, Feb. 2011.

[22] Kai Wang, Jufeng Yang, Guangshun Shi and Qingren Wang, “An Expanded Training Set Based Validation Method to Avoid Overfitting for Neural Network Classifier,” *Journal of 2008 Fourth International Conference on Natural computation*, Vol. 3, pp. 83-87, Oct. 2008.

통계적 가중치를 이용한 협력형 소스측 분산 서비스 거부 공격 탐지 기법

염성웅, 김경백
전남대학교 전자컴퓨터공학부
yeomsw0421@gmail.com, kyungbaekkim@jnu.ac.kr

Collaborative Source-Side DDoS Attack Detection using Statistical Weight

Sungwoong Yeom, Kyungbaek Kim
Dept. Electronics and Computer Engineering, Chonman National University

요 약

최근 보안이 취약한 IoT 기기를 악용하는 분산 서비스 거부 공격을 보다 빠르게 감지하고 공격자의 위치를 확보하기 위해, 소스측 서비스 거부 공격 탐지 기법이 연구되었다. 또한, 대규모 분산 서비스 거부 공격의 효과적인 탐지를 위해 여러 사이트의 소스측 탐지결과를 공유하여 탐지 성능을 향상시키는 협력형 소스측 서비스 거부 공격 탐지 기법도 연구 되었다. 이러한, 협력형 탐지 기법에서는 서로 다른 시간대에서 수행된 공격 탐지 성능 편차에 의해 전체적인 공격 탐지 성능이 영향을 받을 가능성이 있다. 이 논문에서는 공격 탐지 기법의 각 시간대별 탐지 성능을 통계적으로 분석한 가중치를 이용하는 협력형 소스측 분산 서비스 거부 공격 탐지 기법을 제안한다. 특히, 공격을 오탐지하는 거짓 양성률(false positive rate)를 고려하여, 협력형 탐지 기법의 오탐율을 낮출 수 있도록 하였다. 실제 DNS요청 트래픽을 수집하고, 이를 기반으로 실험한 결과, 제안하는 가중치를 사용하였을 때, 공격탐지율은 높게 유지하면서, 오탐율을 약 30% 줄일 수 있음을 확인하였다.

Ⅰ. 서 론

IoT 환경의 활성화에 따라, 이기종 네트워크 엔티티로 구성된 협업 네트워크 시스템(CNS)도 커지는 추세이다. 하지만, 여러 지역에 분포한 IoT 기기의 보안이 취약함을 악용한 분산 서비스 거부 공격의 위협이 급격히 증가하고 있다. 이렇게 악용된 IoT 기기들이 생산한 트래픽의 량은 소량이지만 피공격자 측 네트워크에서는 대량의 공격 트래픽이 유입된다. 또한 피공격자 측 네트워크는 지연된 탐지 및 공격자 추적의 어려움과 같은 몇 가지 단점이 드러난다. 최근 에지 컴퓨팅의 발달됨으로써 소스측에서 네트워크 공격 트래픽 탐지 기법을 위해 다양한 연구들이 진행되고 있다.

소스측에서 발견되는 공격 트래픽의 총량은 피해자 측에서 발견되는 공격 트래픽에 비해 상대적으로 작기 때문에 공격 트래픽이 정상적인 네트워크 트래픽에 쉽게 섞일 수 있다. 이와 같은 공격트래픽을 탐지하기 위해 관찰된 트래픽의 양을 이용하여 동적으로 공격 탐지 임계 값을 변경하는 기법이 연구되었다.[1] 하지만, 특정 지역에 고정되어 있는 소스측 공격 탐지 모듈은 대규모로 분산되는 공격의 특징을 포착할 수 없다. 또한, 소스측 공격 탐지 모듈의 성능은 시차가 위치한 지역의 네트워크 에지에서 관찰되는 시간과 트래픽의 특징에 따라 성능이 다를 수 있다.

이 소스측 공격 탐지 모듈별 성능 편차를 극복하기 위해, 각 소스측 공격 탐지 모듈의 시간 인덱스 별 탐지 결과를 공유하여 최종 탐지 결과를 도출하는 연구가 진행되었다.[2] 이 기법은 공격이 탐지된 모듈의 수가 주어진 임계값보다 높을 경우 공격을 탐지하도록 한다. 하지만, 이러한 기법은 정상적인 트래픽을 공격 트래픽으로 오탐지된 결과들 또한 함께 공유되기 때문에 최종적으로 공유하여 결과를 도출할 때 협력형 공격 탐지 모듈의 오탐율이 증가할 수 있다.

이 논문에서는 여러 사이트에 위치한 소스측 서비스 공격 탐지 모듈의 시간 인덱스 별 탐지 성능을 통계적으로 계산한 가중치와 탐지결과를 공유하여 공격여부를 판단하는 협력형 소스측 분산 서비스 거부 공격 탐지 기법을 제안한다. 각 소스측 공격 탐지 모듈은 적응형 임계를 사용하여 공격을 탐지한다. 소스측 공격 탐지 모듈의 시간 인덱스 별 탐지 성능을 통계적으로 계산한 가중치는 해당 시간 인덱스에 공격이 존재하였을 때 탐지될 확률과 공격이 존재하지 않았을 때 잘못 탐지되지 않을 확률의 합, 즉, 소스측 공격 탐지 모듈이 통계적으로 해당 시간 인덱스에 공격 트래픽과 정상트래픽을 정확하게 분류할 확률로 설정한다. 협력형 공격 탐지 모듈은 서로 다른 시간대에 위치한 소스측 공격 탐지 모듈의 탐지 결과와 가중치들을 공유하고 가중치 산술 평균을 계산한다. 최종적으로 가중치 산술 평균과 임의의 임계값과 비교하여 공격을 판단한다.

제안된 기법의 실효성 검증을 위해, 실제 DNS 트래픽 데이터에 기반한 실험을 수행하여 제안된 기법이 공격탐지율(Detection Rate)과 공격오탐율(False Positive Rate)에서 기존의 공격탐지 기법의 성능을 능가하는 것을 확인하였다.

Ⅱ. 관련 연구

과거에도 서비스 거부공격 탐지를 위한 협력형 공격 탐지 모델은 연구되어 왔다. [4,5,6,7,8,9] 이 연구들은 주로 네트워크 구조를 이해하여 네트워크 트래픽이 공격자가 위치한 네트워크에서 피공격자가 위치한 네트워크로 유입되는 과정에서 발생하는 공격 네트워크 트래픽의 결집 정도 및 유입 정도를 활용하여 분산 서비스 거부 공격을 탐지하는 알고리즘 또는 동적으로 자원 할당을 통해 협업 네트워크를 용이하게 하는 시스템을 제안하고 있다. 그러나, 이 논문들은 공격 네트워크 트래픽의 볼륨이 정상 네트워크 트래픽 볼륨과 확연히 차이나는 상황을 주로 가정하고 있으며, 소스측 보다는 피공격자 측에 가까울수록 탐지가 더 잘되는 알고리즘을 제안하고 있다.

이와 반대로 공격원이나 피해자 네트워크 근처에 위치하여 서비스 거부

공격 탐지를 위한 협력형 공격 탐지 모델 또한 연구되어 왔다. [2,3] 이 연구들은 각 소스측 네트워크에 공격 탐지 모듈을 위치시키고 결과를 공유함으로써 연결된 소스측 네트워크의 시너지 효과를 통해 거짓 경보 발생률을 크게 방지하였다. 하지만, 이러한 기법들은 공격 트래픽으로 오탐지된 결과들이 정상적으로 탐지된 결과들과 함께 공유되기 때문에 최종적으로 공유하여 결과를 도출할 때 협력형 공격 탐지 모듈의 오탐율이 증가할 수 있다.

Ⅲ. 통계적 가중치를 이용한 협력형 소스측 분산 서비스 거부공격 탐지 기법

제안된 협력형 소스측 분산 서비스 거부공격 탐지 기법은 서로 다른 시간대에 위치한 소스측 거부 공격 탐지 모듈에서 탐지된 결과와 탐지 결과 통계 기반 가중치를 활용하여 최종 결과를 도출한다.[2] 이때, 각 소스측 거부 공격 탐지 모듈은 관찰되는 트래픽 기반 적응형 임계값 조정 기법을 사용한다.[1] 이 적응형 임계값 조정 기법은 일정한 시간 간격동안 관찰되는 트래픽의 양을 사용하여 예측한 트래픽 양에 Margin을 더하여 동적으로 공격 탐지 임계 값을 조정한다. 이때, 적응형 임계값 조정 기법이 공격에 대한 영향을 받지 않는다는 가정을 적용한다. i번째 소스측 서비스 거부 공격 모듈의 타임윈도우 t_i에 대한 탐지 결과를 $d_i^{t_i}$이라 한다. 타임윈도우 t_i는 1분 간격으로 구성되며 탐지 결과 $d_i^{t_i}$의 값은 공격이 탐지될 경우 1의 값을 가지고 공격이 탐지되지 않을 경우 0의 값을 가진다.

i번째 소스측 서비스 거부 공격 모듈의 타임윈도우 t_i의 탐지 결과 통계 기반 가중치는 N일 동안 가상의 공격을 부여하였을 때 탐지할 확률 D_{t_i}과 N일 동안 가상의 공격을 부여하지 않았을 때 오탐지할 확률 F_{t_i}들의 비중을 달리하여 더한 값, 즉, 소스측 공격 탐지 모듈이 통계적으로 타임윈도우 t_i에 공격 트래픽과 정상트래픽을 정확하게 분류할 확률로 설정한다. 이때, 확률 D_{t_i}와 확률 F_{t_i}의 정의는 $\sum_{k=1}^{N} \frac{d_i^{t_i-(N-k)*1440}}{N}$로 설정한다. 최종적인 가중치를 계산하는 수식은 아래와 같다.

$$W_{t_i} = \alpha * D_{t_i} + (1-\alpha) * (1-F_{t_i}) \quad (1)$$

여기서 계수 α는 D_{t_i}와 F_{t_i}의 비중을 나타내며 0과 1사이의 일정한 값을 가진다. 계수 α 값이 1에 가까울수록 임의의 타임윈도우 t_i에 공격이 존재하였을 때 공격을 탐지할 확률의 비중을 둔다고 할 수 있다. 우리는 계수 α를 0.5로 설정하여 공격이 존재하였을 때 공격을 탐지할 확률과 공격이 존재하지 않을 때 공격을 탐지하지 않을 확률의 비중을 같게 한다. 가중치 산술 평균은 각 사이트 별 소스측 서비스 거부 공격 모듈의 타임윈도우 t_i에 해당하는 탐지 결과 통계 기반 가중치와 탐지 결과를 공유하고 최종적으로 결과를 판단하기 위해 사용된다. 수식은 아래와 같다.

$$A^t = \sum_{i=1}^{L} \frac{W_{t_i} * d_i^{t_i}}{W_{t_i}} \quad (2)$$

가중치 산술 평균값이 임의로 지정된 임계값 θ 보다 클 경우, 최종적으로 해당 타임윈도우 t에서 공격이 탐지되었다고 판단한다.

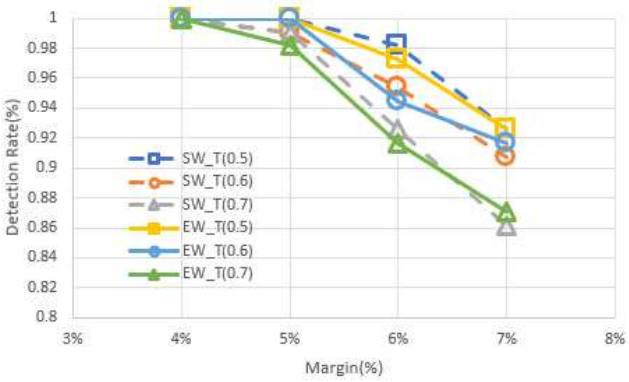


그림 1. Detection Rate for Collaborative DDoS Attack Detection

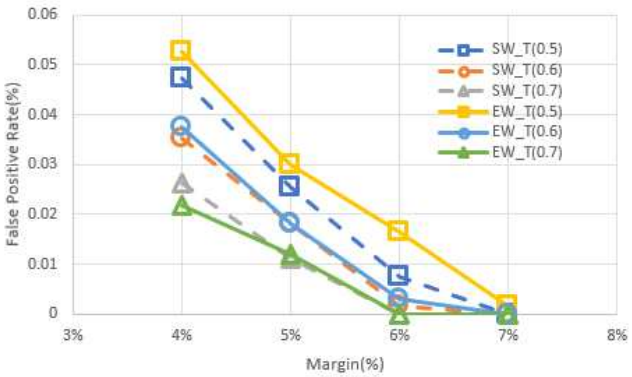


그림 2. False Positive Rate for Collaborative DDoS Attack Detection

Ⅳ. 실험 및 검증

제안된 협력형 소스측 분산 서비스 거부 공격 탐지 기법의 검증을 위해, 서로 다른 시간대에 위치하는 10개의 사이트에서 각각 9일간의 DNS 요청 트래픽을 수집하였다. 수집된 트래픽의 Outlier를 제거한 후, 해당 트래픽을 정상트래픽으로 정의하였다. 각 사이트의 9일간의 DNS 요청 트래픽 중 8일은 가중치를 계산하기 위해 사용되었다. 서비스 거부 공격 트래픽은 해당 트래픽의 마지막 1일에 해당하는 기간에 추가되었으며, 서비스 공격 트래픽은 서로 다른 시간대의 타임 윈도우t_i에서 동시에 발생하도록 하였다. 우리는 서로 다른 시간대에 위치하는 소스측 거부 공격 탐지 기법의 성능과 상관없이 공격 탐지가 된 사이트 수에 결과를 판단한 EW(Equal Weight) 기법[2]과 탐지된 결과 통계 기반 가중치 SW(Statistical Weight) 기법을 비교한다. 이때, 각 소스측 거부 공격 탐지 기법의 임계 값에 영향을 주는 Margin은 4% ~ 7%까지 적용하고, 협력형 공격 탐지 기법의 임계 값θ은 0.1씩 변화 시키면서 테스트한다. 각 소스측 거부 공격 탐지 모듈의 Margin 별각 기법별로 EW 기법과 SW 기법의 공격 탐지율(Detection Rate)과 공격 오탐율(False Positive Rate)를 측정한다.

그림 1의 Margin의 변화에 따른 각 협력형 공격 탐지 기법의 공격 탐지율(Detection Rate)은 각 소스 측 공격 탐지 모듈의 결과와 가중치를 공유하여 최종적으로 결과를 도출하였을 때 전체 분산 서비스 거부 공격 수 중 탐지된 공격 횟수를 뜻한다. 그림 2의 Margin의 변화에 따른 각 협력형 공격 탐지 기법의 공격 오탐율(False Positive Rate)은 각 소스 측 공격 탐지 모듈의 결과와 가중치를 공유하여 최종적으로 결과를 도출하였을 때 정상적인 트래픽 수 중 공격으로 오 탐지된 공격 수를 뜻한다. 그림 1과

그림 2에서 T(x)는 임계값 θ 를 x로 설정하는 것을 의미한다.

실험 결과에서 SW기법은 EW기법과 비슷한 Detection Rate 값을 보이면서도, 전반적으로 낮은 False Positive Rate 값을 가지는 것을 확인할 수 있다. 특히, 완벽한 detection rater를 가지는 설정인 Margin 4% 또는 5%에서 임계값이 0.5 또는 0.6인 경우, SW 기법은 EW 기법에 비해 공격 오탐율을 최대 30%정도 낮출 수 있음을 확인하였다.

Ⅴ. 결론

본 논문에서는 적응형 임계값 조절 기법을 이용하는 소스측 서비스 거부 공격 탐지 기법의 성능을 향상시키기 위한 협력적 기법을 제안하고, 실제 네트워크 트래픽에 이용한 평가를 통해 제안된 기법의 성능을 검증하였다. 검증된 결과로 SW 기법은 EW 기법에 비해 전반적으로 공격 오탐율을 낮추며 최종적으로 30%정도 낮출 수 있음을 확인하였다. 향후, 서로 다른 시간대에 위치한 소스측 공격 탐지 모델의 Margin 값을 관찰되는 트래픽 량과 특징에 따라 동적으로 수정하는 기법을 제안하고자 한다.

ACKNOWLEDGMENT

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2017RIA2B4012559). 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음" (IITP-2020-2016-0-00314).

참 고 문 헌

[1] Nguyen, Sinh-Ngoc, Nguyen, Van-Quyet, Nguyen, Giang-Truong, Kim, JeongNyeo, Kim, and Kyungbaek, "Source-Side Detection of DRDoS Attack Request with Traffic-Aware Adaptive Threshold." IEICE Transactions on Information and Systems 101.6 (2018): 1686-1690.

[2] 염성웅, and 김경백. "협력형 소스측 서비스 거부 공격 탐지 기법 연구." 한국통신학회 학술대회논문집 (2019): 478-479.

[3] Song, ByungHak, Heo, Joon, and Hong, Choong Seon, “Collaborative Defense Mechanism Using Statistical Detection Method against DDoS Attacks.” IEICE Transactions 90-B (2007): 2655-2664.

[4] Shalinie, S. Mercy, et al. "CoDe－An collaborative detection algorithm for DDoS attacks." 2011 International Conference on Recent Trends in Information Technology (ICRTIT). IEEE, 2011.

[5] DChen, Yu, Kai Hwang, and Wei-Shinn Ku. "Collaborative detection of DDoS attacks over multiple network domains." IEEE Transactions on Parallel and Distributed Systems 18.12 (2007): 1649-1662.

[6] Chen, Yu, and Kai Hwang. "Collaborative change detection of DDoS attacks on community and ISP networks." International Symposium on Collaborative Technologies and Systems (CTS'06). IEEE, 2006.

[7] Tariq, Usman, et al. "Collaborative peer to peer defense mechanism for ddos attacks." Procedia Computer Science 5 (2011): 157-164.

[8] Rashidi, Bahman, Carol Fung, and Elisa Bertino. "A collaborative DDoS defence framework using network function virtualization." IEEE Transactions on Information Forensics and Security 12.10 (2017): 2483-2497.

[9] Rashidi, Bahman, and Carol Fung. "CoFence: A collaborative DDoS defence using network function virtualization." 2016 12th International Conference on Network and Service Management (CNSM). IEEE, 2016.

[10] Nguyen, Giang-Truong, Nguyen, Van-Quyet, Nguyen, Huu Duy, and Kim, Kyungbaek,. "LSTM based Network Traffic Volume Prediction.", In Proceedings of 2018 KIPS Spring Conference, 2018.

양자암호 네트워크 표준 및 시스템 구조 연구

상의정, 박춘걸
KT 융합기술원
uijeong.sang@kt.com, peter.park@kt.com

A Study of Standards and System Architecture for Quantum Cryptographic Network

Uijeong Sang, Choongul Park
KT Institute of Convergence Technology

요 약

양자 컴퓨터의 등장으로 기존 네트워크 보안기술에 대한 안정성이 위협받고 있다. 이러한 보안 위협에 대한 해결책 중 하나로 양자키 분배(Quantum Key Distribution, QKD) 기술이 연구되었다. QKD 기술은 양자물리의 중첩성, 불확정성, 비가역성 등 특성에 기반하여 복제가 불가능한 대칭형 암호키를 생성하고 해킹에 대한 탐지를 용이하게 하는 기술로, 수학적 문제를 기반으로 하는 암호화 기법이 아니기 때문에 양자 컴퓨터의 컴퓨팅 파워가 상승하는 것과 관계없이 항상 동일한 안전성을 제공할 수 있다. 그러나 QKD는 1:1 양 종단에 연결된 장비로부터 양자키를 생성하며 50km 이내의 짧은 통신 거리를 갖는 등의 제약으로 QKD 자체만으로 암호통신 네트워크를 구성하는 것이 어렵다. 본 논문은 이러한 문제를 해결하기 위해 연구되고 있는 양자암호 네트워크 표준 및 시스템 구조에 대해 알아본다.

Ⅰ. 서론

오늘날 다양한 서비스가 네트워크를 통해 제공되고 있으며, 특히 금융, 국방, 의료 등 정보보호의 중요성이 높은 분야에서 네트워크 보안과 암호통신의 중요성이 높아지고 있다. 현재 암호통신에서 사용되는 암호화 알고리즘들은 지금의 컴퓨팅 파워로는 암호키를 알아내거나 평문을 알아내는데 연산에 매우 오랜 시간이 소요되기 때문에 안전이 보장되고 있다. 그러나 양자 컴퓨터의 등장과 발전으로 인해 기존의 공개키 암호화 방식은 다항식 시간 내에 해결이 가능하며 대칭형 암호의 무차별 대입 공격 속도 또한 가속될 수 있게 되었다[1].

양자암호키분배(QKD; Quantum Key Distribution)기술은 양자 컴퓨터의 보안 위협에서 안전한 암호통신 기술로서 멀리 떨어진 두 네트워크 노드 간 양자역학의 법칙들을 따르는 대칭형 암호키를 주고 받을 수 있는 기술이다. 그러나 QKD 는 1:1 양 종단에 연결된 장비로부터 양자키를 생성하며 연결거리가 50Km 내외로 기존의 광통신에 비해 짧아 QKD 만으로는 네트워크를 구성하기가 어렵다. 따라서 이러한 QKD 기술을 이용하여 다자간 암호통신을 할 수 있는 양자암호네트워크 상용화 기술개발이 국내외에서 활발히 진행되고 있다.

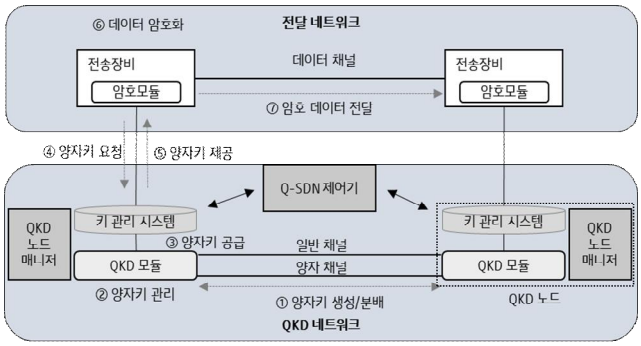
본 논문는 이러한 QKD 의 특성을 고려한 양자암호 네트워크 기술과 관련 표준화 현황을 소개하고 나아가 양자암호 네트워크에서 QKD 에서 생성된 양자키의 관리, 중계, 전달과 QKD 관제 등을 담당하는 양자키 관리 시스템(QKMS; Quantum Key Management System)의 구성과 역할, 기능 등에 대해 알아본다.

Ⅱ. 양자암호 네트워크 기술

QKD 는 1:1 형태의 연결로만 암호키를 제공하고 짧은 연결 거리를 갖기 때문에 QKD 를 그대로 실제 보안 서비스에 적용하는 것은 매우 제한적이다. 따라서 양자암호 네트워크의 형태로 구축하여 QKD 가 갖고 있는 단점들을 극복하고 서비스를 제공하기 위해 양자암호 네트워크 기술이 필요하며 관련 표준들이 제정되고 있다.

양자암호 네트워크는 구현과 표준화 방향에 따라서 용어와 포함하는 기능에 차이는 존재할 수 있으나 양자 계층, 키 관리 계층, 네트워크 제어 계층으로 구분할 수

있다. 양자 계층은 양자의 성질을 활용하여 QKD 장비를 통해 양 종단간 공유되는 랜덤 비트를 생성한다. 키 관리 계층은 양자 계층에서 생성된 랜덤 비트를 전달 받아 실제 서비스에 활용할 수 있는 양자키로 가공, 2 홉 이상의 QKD 연결로 이어진 경우 양자키를 제공하기 위한 키 릴레이, 가공된 양자키를 암호 서비스에 전달 및 파기 등 양자키의 관리 기능을 제공한다. 마지막으로 네트워크 제어 계층은 양자 계층과 키 관리 계층의 정보를 모니터링하고 제어하는 기능을 수행한다.

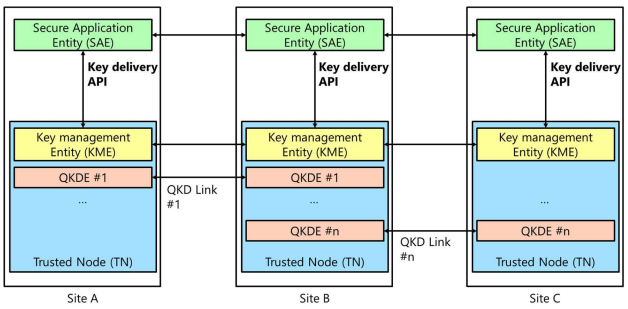


[그림 1] 양자암호 전달 네트워크

그림 1 은 TTA 에서 제정한 QKD 기술과 전달 네트워크(Transport Network) 기술을 결합한 양자암호 전달 네트워크 기능구조[2]에 대한 표준이다. 이 표준은 전달 네트워크에서 양자암호를 사용함에 있어서 필요한 기능요소, 운영절차, 보안 고려사항 등을 제안하고 있다. 전송 장비와 QKD 장비는 함께 신뢰 공간에 존재할 수 있으며, 기존의 전달 채널에 양자 채널을 함께 수용할 수 있다는 장점이 있다. 이와 같이 양자암호 네트워크는 QKD 와 특성과 관련 서비스를 고려한 형태로 구성할 수 있으며 아래에서 소개할 표준화 동향 또한 양자암호 네트워크 표준들은 이를 위한 네트워크 구조, 키 관리 기술, SDN 기술, 인터페이스 등에 대해 다루고 있다.

Ⅲ. 양자암호 네트워크 표준화 동향

ETSI(European Telecommunications Standards Institute, 유럽전기통신표준협회)는 양자키 분배에 사용되는 장비의 인터페이스, 프로토콜, 보안 증명 등 QKD 장비의 정의로부터 시작하는 장비 중심의 표준화를 검토하였다. 양자암호 네트워크는 QKD 에서 생성되는 양자키를 암호화 장비에 전달하는 과정이 필수적이며 ETSI GS QKD 014[3] 표준 문서에서 Quantum Key Distribution(QKD): Protocol and data format of REST-based key delivery API 라는 제목으로 양자키를 관리하는 KME(Key Management Entity)와 양자키를 사용하는 SAE(Secure Application Entity)간의 키 전달 API 를 표준화하였다.

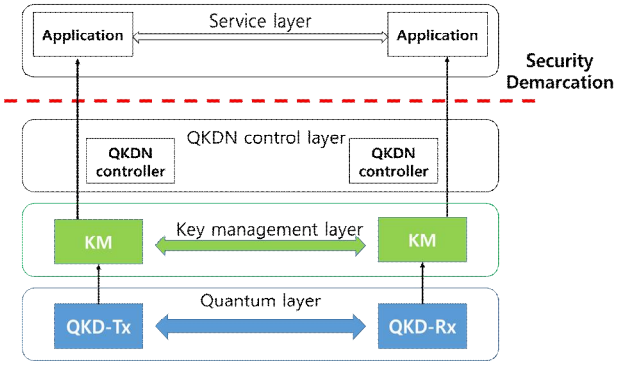


[그림 2] ETSI GS QKD 014 QKD 네트워크 예시

ETSI GS QKD 014 표준은 현재 QKD 장비의 실증에 주로 사용되는 표준으로서 HTTPS 를 통신 프로토콜로 사용하며, 양자키 요청과 양자키 전달에 필요한 정보를 JSON 형태로 전달하는 방식을 사용한다. 그러나 HTTPS 의 암호화 방식이 양자 컴퓨터에 취약하며 키 전달 API 를 사용하는 SAE 또한 안전한 구역에 존재해야 한다.

ITU-T 에서는 QKD 와 QKD 네트워크의 활용, 구조, 요구사항 및 양자 난수발생기 구조 등에 대한 표준화가 진행 중에 있다. ITU-T 의 국제표준화는 KT(SG13)와 SKT(SG17)에 의해 주도적으로 이루어지고 있으며 QKD 로부터 생성되는 양자키를 사용자 네트워크에 적용시키기 위한 QKD 네트워크의 구조, 제어, 요구사항 등 양자암호 네트워크와 관련된 표준이 다수 포함되어 있다.

SG13 은 네트워크 관점에서 QKD 를 지원하는 네트워크를 위한 프레임워크에 대한 표준 Y.3800[4]을 제안하였고, 이는 QKD 네트워크와 사용자 네트워크간의 관계, QKD 네트워크 요구사항, 구조, 기능을 포함하고 있다. 제안하는 표준은 QKD 네트워크를 네 가지 계층(그림[3] 참조)으로 구분하고 있다. Quantum layer 는 QKD 링크로 연결된 각 쌍의 QKD 장치들을 통해 대칭 랜덤 비트 문자열을 생성한다. 생성된 랜덤 비트 문자열은 동일한 QKD 노드에 위치한 KM(Key Manager)에 전달된다. 또한 QKD 장치는 QKD 링크 패러미터를 QKDN 매니저에 전달한다. Key management layer 는 키의 생성, 파기와 같은 관리 기능과 키 제공을 위한 인터페이스를 포함하고 있다. Key management layer 는 동일 QKD 노드에 존재하는 QKD 로부터 제공받는 랜덤 비트 문자열에 대한 동기화와 재가공을 통해 사용자가 사용할 수 있는 형태의 양자키로 생성하고 제공한다. QKDN control layer 는 각 QKD 노드에서 키 중계를 위한 라우팅 컨트롤, 양자 링크 컨트롤, QKD 서비스 세션 컨트롤 등의 QKD 네트워크 컨트롤 기능을 제공하며 QKDN management layer 는 장애, 설정, 성능, 보안등을 포함하는 전체 QKD 네트워크의 모니터링과 관리를 수행한다.



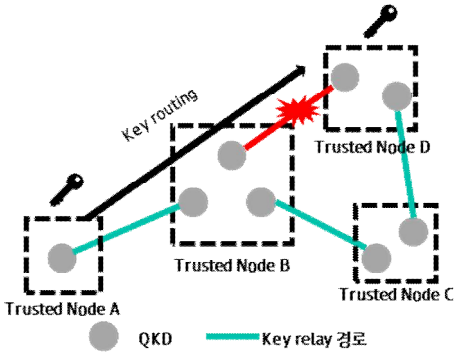
[그림 3] Y.3800 QKD 네트워크와 사용자 네트워크 구조

Y.3800 은 그림 3 과 같이 QKD network 와 User network 사이의 보안 경계(Security demarcation)를 통해 양자키와 네트워크 관리 기능의 요청, 제공, 공급을 위한 공통 인터페이스를 제공하도록 정의 하였다. 또한 보안 경계는 보안 요구사항과 기능을 충족하여야 한다. QKD 에서 양자의 특성을 활용하여 생성되는 양자키는 정보이론적으로 해킹에 안전하나 신뢰구간인 QKD 노드를 벗어나 보안 경계를 통과하여 사용자 네트워크로 양자키가 제공되면 기존 네트워크의 보안이 적용되기 때문에 이러한 보안 문제에 대한 논의가 필요하다.

Ⅳ. 양자암호 네트워크 시스템 구조

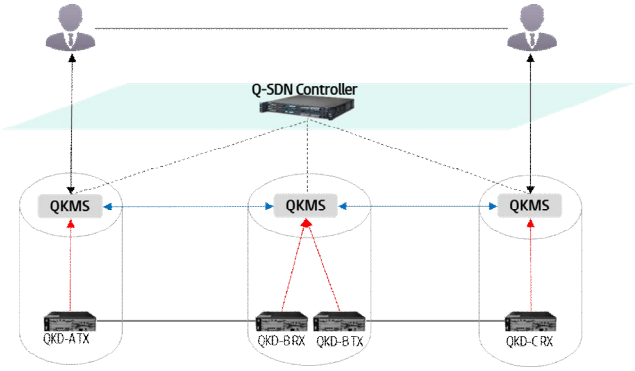
양자암호 네트워크에서 QKD 는 대국의 장비와 연결된 양자 채널과 일반 채널을 통해 대칭형 양자키를 생성 및 분배한다. 생성된 양자키는 사용자의 요구 사항에 맞는 형태로 가공되어 제공되고 이를 통해 종단 간 양자암호 통신 서비스가 제공된다. QKD 양 종단간의 양자암호 통신과 달리 양자암호 전달 네트워크는 네트워크 운용을 위한 다양하고 복잡한 기능을 제공하기 위한 양자키 관리 시스템이 필수적이며, 이는 QKD 와 분리되어 개발 및 운용되어야 한다.

양자키 관리 시스템은 QKD 와 사용자 사이에서 양자키를 암호통신에 요구되는 형태로 제공하는 기능을 포함하여, 양자키의 생성, 삭제 등 양자키 관리 등 양자암호 통신 서비스를 위한 기능들을 제공한다. 나아가 양자키 중계 경로 재설정, 다기종 QKD 운용, 장애 상황 대처 등의 네트워크를 위한 고도화된 기능을 제공한다. 이 절에서는 양자암호 네트워크에서 QKMS 가 제공하는 기능과 역할에 대해 기술한다.



[그림 4] QKMS 간 양자키 중계를 통한 장애 경로 절체

그림 4 와 같이 다수의 QKD 를 사용하여 구성된 양자암호 네트워크의 경우, 물리적으로 보안이 확보된 신뢰 노드(Trusted Node)[5]에 QKD 가 존재한다. 양자암호 네트워크는 신뢰 노드 사이에 직접적으로 연결된 QKD 가 존재하지 않으면 반드시 양자키 중계가 필요하다. 그림 4 의 경우, 신뢰 노드 A 에서 신뢰 노드 D 로 양자키를 중계하는 최단 경로 A-B-D 에서 장애가 발생하면 양자키 관리 시스템은 다른 가용 경로인 A-B-C-D 를 통해 양자키를 중계한다. 예시와 같이 QKD 의 장애 상황 발생 시에도 끊임없는 서비스 제공을 위해 양자키 관리 시스템은 각 신뢰 노드에 존재하며 QKD 의 양자키 생성 정보, 연결 정보를 수집하고 중계 경로에 따라 양자키를 중계한다. 또한 신뢰 노드 양 종단에서 양자키 제공 요청 시 양자키 동기화 문제 등 양자키를 서비스로서 제공하기 위한 기본적인 기능들을 포함하고 있다.



[그림 5] QKMS 연동 인터페이스

양자키 관리 시스템은 양자암호 네트워크에서 서비스 제공과 네트워크 제어 및 관리를 위해 네트워크 구성요소와의 연동 인터페이스과 프로토콜을 제공한다. QKD 를 통해 생성되는 양자키를 공급받고 QKD 에 대한 관제를 수행하기 위한 QKD-QKMS 간 연동 프로토콜과 양자키를 서비스로서 제공하기 위한 QKMS-QKMS 키 관리 프로토콜, ETSI GS QKD 014[3] 표준과 같은 양자키 제공 프로토콜, SDN 을 사용하는 중앙화된 양자암호 네트워크 제어를 위한 QKMS-SDN 연동 프로토콜을 통해 양자암호 네트워크의 핵심이 되는 QKD 에 직접 접근하는 일 없이 각각의 신뢰 노드를 제어하며 양자암호통신 서비스를 제공한다.

Ⅴ. 결론

양자암호통신 인프라 기술이 기초적인 연구단계에 벗어나 상용화 논의가 국내외에서 활발하게 진행되고 있다. 이미 영국, 미국, 중국, 일본 등에서 양자 암호통신 네트워크가 구축되었고, 금융, 행정, 제어 네트워크 등 신뢰할 수 있는 인프라를 필요로 하는 도메인을 중심으로 다양한 서비스 개발을 진행 중이다. 특히 ETSI, ITU-T 뿐만 아니라, 사실상 표준을 주도하고 있는

IETF 에서까지 기존 네트워크와 양자 네트워크를 결합한 인프라에 필요한 표준화 기술에 대한 논의를 진행하는 것은 주목할 만한 변화로 보여진다.

본 논문은 현재 활발히 연구되고 있는 양자암호 네트워크에 대한 국제 표준 기구의 표준화 방향성과 QKD 의 종단간 양자키 분배 기술을 네트워크의 형태로 구성하기 위해 QKD 로부터 전달된 키를 관리(생성, 사용, 폐기)하고, 신뢰 노드에서 QKD 관리를 수행하는 양자키 관리 시스템을 포함한 양자암호네트워크 시스템 구조에 대해 알아보았다. 양자암호통신 서비스의 활성화를 위해 앞으로도 양자암호 네트워크 구축을 위한 더 다양한 기술에 대한 연구와 개발이 이루어질 것이라 생각된다.

참 고 문 헌

[1] Mosca, Michele. "Cybersecurity in an era with quantum computers: will we be ready?." IEEE Security & Privacy 16.5 (2018): 38-41.

[2] TTAK.KO-01.0214, “양자암호 전달 네트워크 기능구조”, 2019.12.

[3] ETSI GS QKD 014 V1.1.1, “Quantum Key Distribution(QKD); Protocol and data format of REST-based key delivery API”, 2019.02.

[4] ITU-T Y.3800, “Overview on networks supporting quantum key distribution”, 2019.10.

[5] ETSI GS QKD 007 V1.1.1, “Quantum Key Distribution (QKD); Vocabulary”, 2018.12.

한국통신학회 통신망운용관리연구회
2020년 통신망운용관리 학술대회 논문집
Proceedings of KNOM Conference 2020

ISSN : 2586-0232 (Online)

2020년 5월 15일 인쇄
2020년 5월 15일 발행

발행인 / 석우진 운영위원장
편집인 / 주홍택 출판위원
발행처 / 한국통신학회 통신망운용관리연구회
서울시 서초구 서초동 1330-8 현대기림오피스텔 1504동 6호
전　　화: 02-3453-5555
홈페이지: www.knom.or.kr
디자인 및 편집 / 계명대학교 COMNET

한국통신학회 통신망운용관리연구회